CASE INSTITUTE OF TECHNOLOGY

CLEVELAND, OHIO

ENGINEERING DESIGN CENTER

This Research Was Sponsored By

THE NATIONAL AERONAUTICS AND SPACE ADMINISTRATION

A Two-Dimensional Absolute Digital Data Path Control System

Report No. EDC-1-63-18 OTS: $11.50 ph, $4.79 mf

by

William H. Ninke

Harry W. Mergler
Professor of Engineering
Principal Investigator
(NASA Grant NsG-36-60)

Digital
Systems
Laboratory

September 1963 153p refs

ABSTRACT

11105

A two-dimensional path generator which produces absolute digital interpolated path control information from discrete input path points is designed, implemented and tested. The path points are specified at equal intervals of one of the variables. A central interval interpolation scheme is used, i.e. the path in each interval is a section of the third order polynomial which passes through four successive input path points including the end points of the interval, the previous path point, and the next path point. The polynomials are generated by three connected digital integrators operating upon the respective derivatives of the polynomials. Simple corrections to two of the integrators allow changing from the polynomial appropriate in one interval to that appropriate to the next. Provisions are included which allow the generation of both single valued paths and closed contours. A scaling and error analysis of the path generator are made allowing given performance criteria to be met.

A two-dimensional plotter is constructed which converts the digital outputs of the generator into a graphical record. The combination of the generator and the plotter represents a path control system. Various path records of this system along with examples of the input data preparation are presented. AUTHOR

ACKNOWLEDGEMENTS

The author wishes to express his appreciation to his advisor, Professor Harry W. Mergler, who has provided considerable help and encouragement during the past four years. Thanks are due to Mrs. Janet Kicher, Mr. Chauncey Hoyt and Mr. David Lee who aided in various stages of this project. Special thanks are due to the author's fellow members of the Case Digital Systems Laboratory who contributed both aid and inspiration.

The support of this project by the National Aeronautics and Space Administration is gratefully acknowledged.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF SYMBOLS

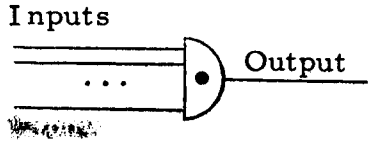
a	Polynomial coefficient
A	Binary number
A_i	The ith bit of A
A^1	One's complement of A
B	Binary number
B	Leading brush for V-brush readout
B_i	The ith bit of binary number B
$\bar{B}$	Lagging brush for V-brush readout
C_i	Carry input bit to the ith stage of a binary adder
E	Total output error of a digital integrator
E_r	Roundoff error of a digital integrator
E_t	Truncation error of a digital integrator
E_s	Supply voltage
$f(x)$	Function of x
$F(x)$	Indefinite integral of f(x)
h	Fixed interval of a coordinate direction
i	Index integer
K_2	Initial value of the second derivative
K_3	Initial value of the third derivative
m	Index integer
n	An integer

N	An integer
p	Binary number
R	Reset input of a flip-flop
S	Set input of a flip-flop
S	Binary sum number
S_i	The ith bit of number S
T	Trigger input to a flip-flop
V_0	Output voltage of comparator
V_{0L}	Output of voltage ladder decoder
V_{0C}	Output voltage from carry function
x	A Cartesian coordinate
y	A Cartesian coordinate
α	Interpolation parameter
β	Error parameter
δ	Least significant bit of an integrand register
Δ	Difference operator
Δx	Increment in x

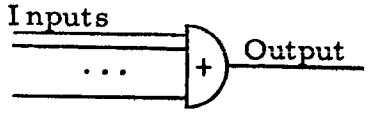
Logical Connectives

$\cdot$	(with or without the dot) AND
+	OR
$\oplus$	EXCLUSIVE-OR
|	NAND

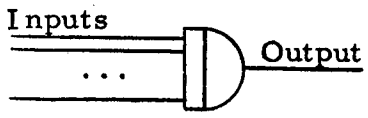
Logical Element Symbols



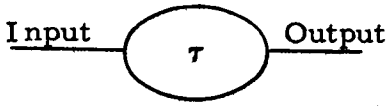
AND



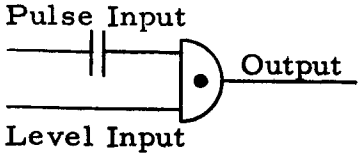
OR



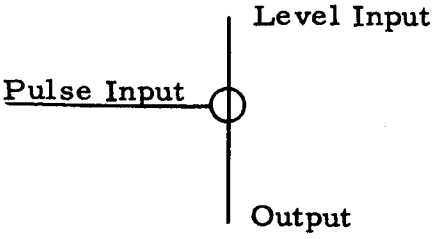
NAND



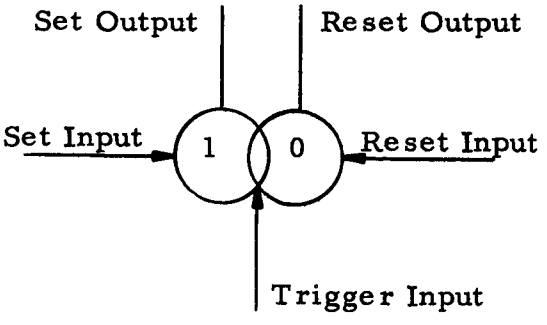
Delay



Gated Pulse Generator



Gate



Flip-Flop

CHAPTER I

PATH CONTROL SYSTEMS

Introduction

The purpose of a path control system is to control precisely and continuously the relationship between the output variables of the system. The desired relationship is previously determined and is usually supplied as a limited amount of input data. Generally the variables represent positions although this is not a binding restriction. Definite power requirements are inherent in the control of the variables. Although the maintenance of the relationship between the output variables at all times is the primary requirement, depending on the particular application, the maintenance of a given path velocity or a component velocity may also become important.

The question arises how an effective path control system might be achieved. An answer that has been widely accepted is the use of the outputs of an accurate path generator as the inputs to high performance power servomechanisms, one servo being used for each variable. Such a system is shown in Fig. 1.1. The use of high resolution feedback elements and high gains in the servos allows accurate following of the outputs of the path generator by the outputs of the servos.

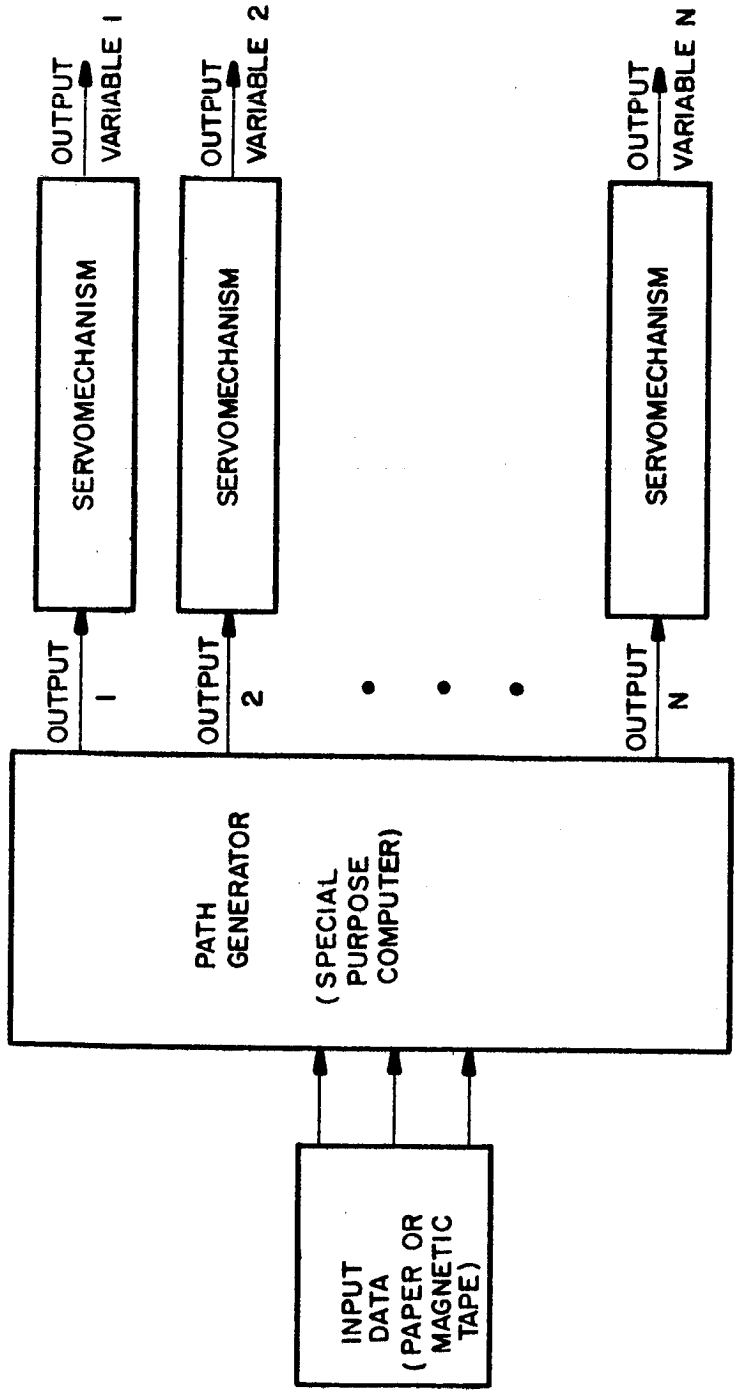


FIGURE I.I PATH CONTROL SYSTEM

A system of the form of that shown in Fig. 1.1 is the subject of this research. Before details of this are discussed, however, an enumeration of the advantages of the use of digital techniques in such a system and a discussion of the components of the system will be presented.

Digital Techniques

Early path control systems were based upon analog methods and components.[13,14,18,26]* Such systems, however, had inherent restrictions. To remove these restrictions and to achieve additional advantages, later systems employed digital techniques.[4,7,15,21]

In a digital system, a variable is measured by the state of a circuit or a transducer, not by the magnitude of an output signal as is the case with analog circuits or transducers. This type of measurement or representation reduces the effects of noise and greatly extends possible accuracy limits. Resolution is dependent on the number of discrete digital increments assigned to the range of a variable. In analog systems, the resolution of each component must be increased. This is limited by the state of the art in component manufacture and also by the prohibitive costs of

---

*The superscript numerals refer to the Bibliography.

extremely precise components. Another advantage realized by digital systems with particular importance in the area of the path generator is that accuracy and resolution do not degenerate with transmission or manipulations, since linearity in amplification and preservation of precise wave forms are not essential. Thus during calculations or manipulations non-degenerative storage can be easily provided.

Because of these advantages, digital techniques will be used in both the path generator and following servomechanisms of the system which is described in this report.

Path Generator

The path generator operates upon the limited amount of input data to supply continuous real time information to the servos. If the data is in the form of distinct path points, the path generator performs interpolation. If the input data is in the form of mathematical equations of the path, the generator performs a continuous evaluation. Data in the form of path points or equations can usually be obtained by hand calculation or limited machine calculation. The path generator is actually then a special purpose computer which completes the processing of the input data.[10]

The choice could be made of having the servos controlled directly by a general purpose computer with both the initial data

processing and any interpolation or evaluation being performed there. However, this means engaging a general purpose computer continuously with a resultant high cost of operation. Thus, for economic reasons, the compromise is made that any extensive input data processing is quickly done in a general purpose computer and the real time processing is handled by the path generator leaving the general purpose machine free to perform other tasks.

Up to the present time, only incremental digital techniques have been used in path control systems. In such techniques, a pulse is generated by the path generator on an output variable line for each desired increment of movement in the following servomechanism of that variable. Both positive and negative increment pulses can, of course, be generated.

If instead of providing a pulse for each desired increment of movement of a variable and thus defining any changes with respect to the previous value, a path generator could also operate in a fashion that the absolute position number of each variable is presented, this absolute number changing as the value of the coordinate changes along the path to be traversed. Thus all values are referenced to absolute points, and not to just the previous points.

The two forms of output information available from the path generator, i.e. incremental or absolute representation, dictate the type of servo which must be used to accept the data.

Digital Servomechanisms

A digital servomechanism is one in which control signals in one or more portions of the system are expressed in a numerical code. There are two basic classifications for digital servo systems corresponding to the form of the input and feedback information: absolute systems and incremental systems.[16]

A simple incremental servo system for one variable is shown in Fig. 1.2. The input data is in the form of direction sensitive pulses, each of which represents a desired incremental change of the controlled variable. The time between input pulses is indicative of the desired rate of change of the output. A quantizer, a direction sensitive device which emits a pulse whenever the controlled variable takes on specified values, is the transducer used in the feedback path. Comparison is accomplished by a bidirectional counter (BDC) which accepts both input and quantizer pulses in such a manner as to count up for positive transitions of the input and negative transitions of the controlled variable, and count down for negative inputs and positive changes of the output. Thus, the resultant count in the counter equals the

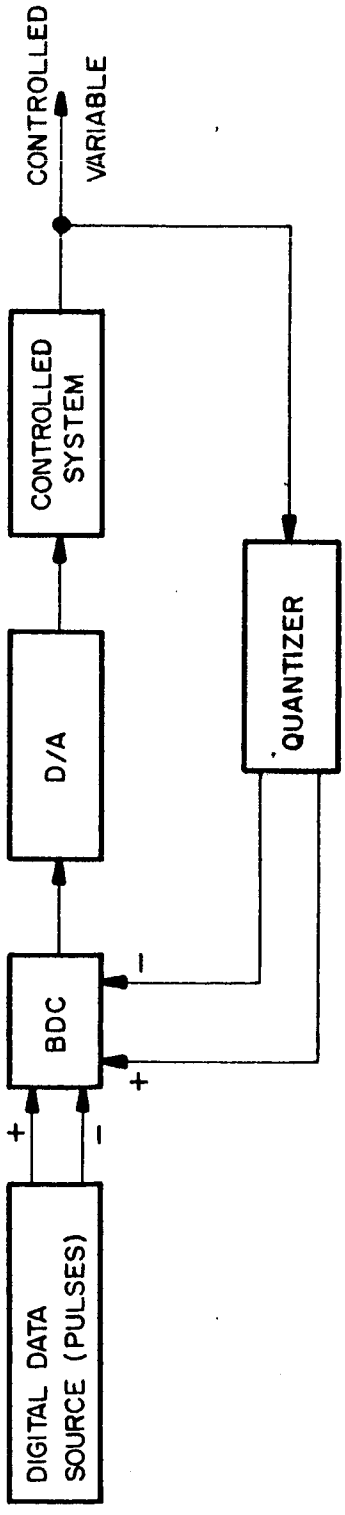


FIGURE 1.2 SIMPLE INCREMENTAL DIGITAL SERVOMECHANISM

digital error in increments. The corresponding analog signal which is produced by the digital to analog converter (D/A) directs the controlled system changing the controlled variable to reduce the error.

To obtain path control, a train of pulses is supplied from the path generator and applied to the input of the bidirectional counter. The number of pulses supplied represents the desired change in position in increments and the manner in which they enter the system represents the path.

A block diagram of a simple absolute system for control of one variable is shown in Fig. 1.3. Input data, the desired value of the controlled variable, is in the form of absolute coded numbers. An encoder in the feedback path converts the controlled variable into another coded number. A comparator produces the difference of these numbers, the digital error. The corresponding analog signal which is produced by the digital to analog converter (D/A) directs the controlled system, changing the controlled variable to reduce the error.

To provide path control, one must supply as input data the coded numbers representing closely spaced points on the desired path. These points are supplied as a function of another

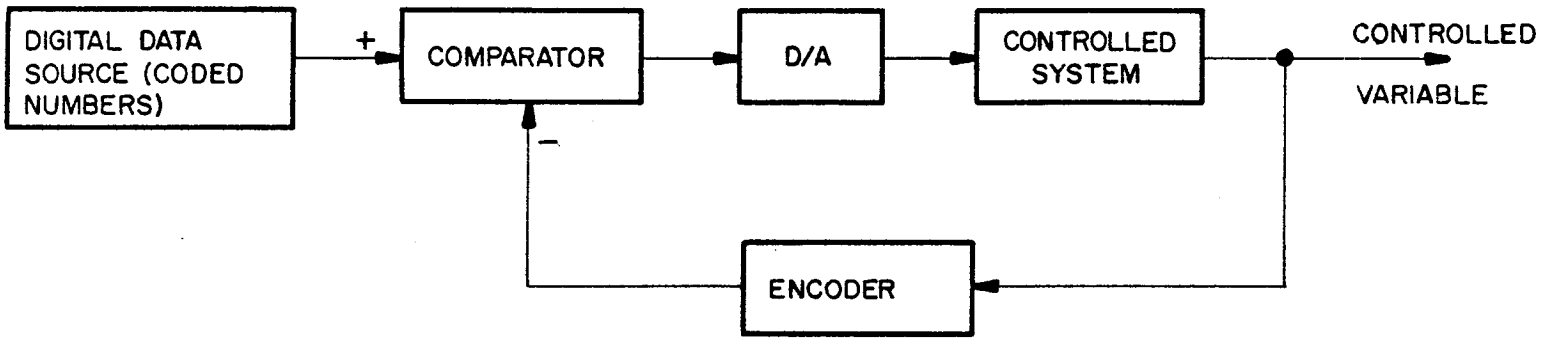


FIGURE 1.3 SIMPLE ABSOLUTE DIGITAL SERVOMECHANISM

variable such as time or a second system variable. The closer the points are located, the better the approximation is to the desired path.

Comparison of Incremental and Absolute Digital Servomechanisms

The following servomechanisms associated with a path control system are usually subjected to severe environmental conditions. Therefore, reliability in these is of great importance.

In an incremental digital servomechanism, pulses might be added or lost through an intermittent component failure or noise introduced by supply or line voltage fluctuations. These errors may cancel one another but there is a greater probability that they will accumulate. Since each new change is based on the previous value, accumulated errors can cause unsatisfactory final results. Also in an incremental servo system, power failure results in loss of reference position. The whole process must be started over, or at least must be referred back to some check point.

An absolute digital servomechanism may also have spurious errors introduced. However, since such a system is based on an absolute reference, errors do not accumulate. Any stoppage through a power failure or other interruption does not cause serious problems in resynchronizing the servo with the input data

since the point of interruption can be returned to through use of the absolute reference and operation continued.

From the above discussion, it can be seen that there are advantages in the use of absolute digital servomechanisms. However, these have not been used in path control applications due to the lack of satisfactory path generators to supply continual absolute digital path information to the servos. Therefore, this research is concerned with the design, construction, and evaluation of an absolute digital data path generator which is used to control two absolute digital data servo systems, the combination serving as a two-dimensional path control system.

The generator is designed to fit a "smooth" function through discrete path points which are spaced at equal intervals of one of the variables. This function consists of interlaced sections of third order polynomials. The polynomials are generated by three connected digital integrators operating upon the derivatives of the polynomials. Thus the path generator might be considered as a third order interpolator. Actual input data is not the path points but, once initial conditions are set, consists only of fourth differences of the values of the dependent variable at the discrete path points.

Design details and final performance records of the system may be found in the following body of this report. Actual construction details are considered in Appendices II and III.

CHAPTER II

THE PATH GENERATOR

Method of Path Generation

The two-dimensional generator of this research has been designed to perform as an interpolator, i.e. it produces a "smooth" function from discrete path points. These path points are specified in a Cartesian system by x and y position variables. The only restriction upon the path points is that they be spaced at equal intervals of one of the variables. This restriction has been placed because, as will be seen, it results in a considerable simplification of equipment and of input data preparation. It is not a serious restriction since, as also will be seen, changes in scaling and in which of the two variables has the equally spaced requirement can be made as different portions of the path are generated. Thus quite a wide variety of spaced points can serve as inputs. To avoid later confusion, the nomenclature is established that the variable along which the points are equally spaced is called the independent variable and the remaining one, the dependent variable.

The "smooth" function connecting the discrete points is formed by interlacing third order polynomials. To see how this is accomplished, consider the path points shown in Fig. 2.1.

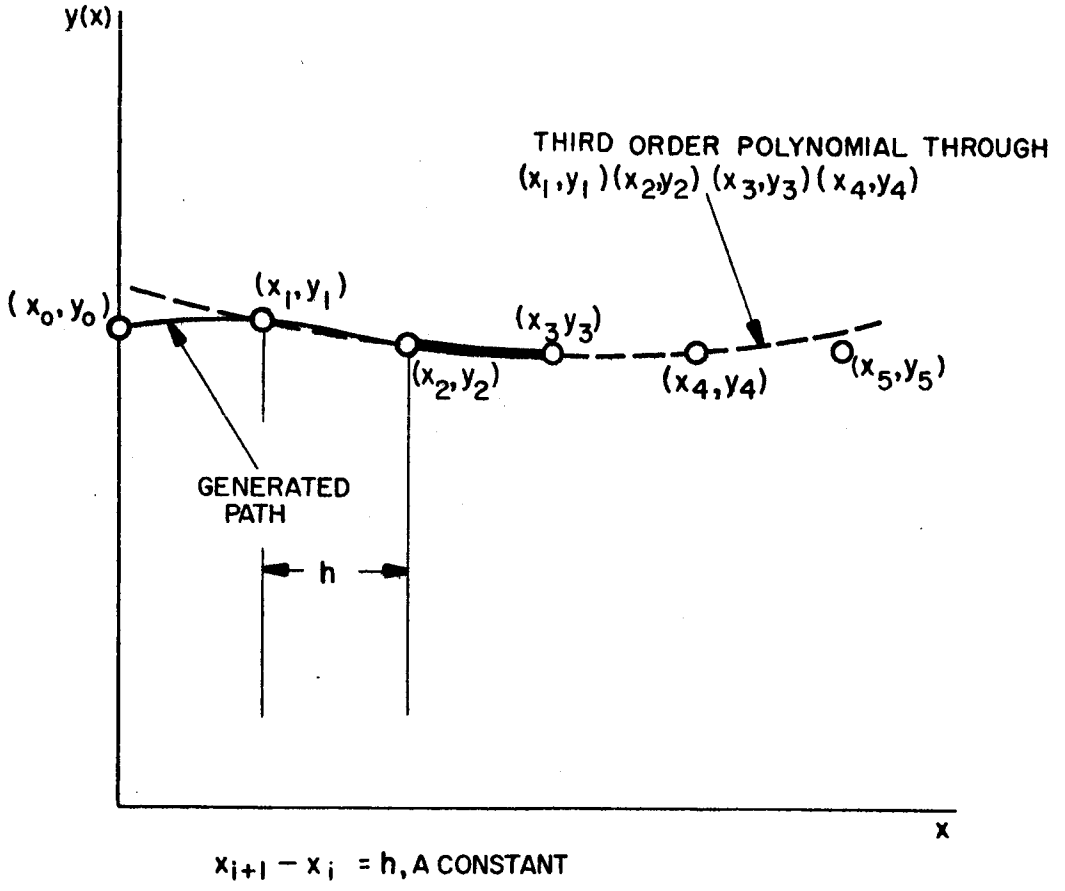


FIGURE 2.1 PATH GENERATION

Here x is the independent variable along which path points are supplied at equal intervals h. Assume that the path has been generated up to the point (x_2, y_2). To produce the portion between (x_2, y_2) and (x_3, y_3), a third order polynomial is fitted through the four points (x_1, y_1), (x_2, y_2), (x_3, y_3) and (x_4, y_4). Then only the central section of the polynomial, that between (x_2, y_2) and (x_3, y_3), is actually generated. To produce the section between (x_3, y_3) and (x_4, y_4), a third order polynomial is fitted through the four successive points beginning with (x_2, y_2) and again only the central section is actually generated. This process of fitting a third order polynomial through four data points and then actually using only the central section is continued to produce the total path. The use of only the central section gives both a look ahead and a look back feature.

The method just described is used with slight modifications in generating other than single valued functions and in generating discontinuous functions. Equipment limitations dictate modifications under certain other conditions. These modifications along with starting and terminating procedures are discussed later.

The "smooth" function generated can have discontinuities at the point of change from one section to the other so that it is not mathematically smooth. The discontinuities become smaller as

the points fall closer to a continuous third order curve. Also since the function is generated digitally there is an inherent quantization effect. This quantization effect can be seen in the output records shown in Chapter IV.

Since the path is to consist of interlaced third order polynomials, the problems arise as to how these might be generated and also how corrections might be introduced to the generator to change from the polynomial being generated in one section to the one which is required in the next section. These problems will now be discussed.

Generation of a Third Order Polynomial

A third order polynomial, $y(x)$, can be written as

$$y(x) = a_3x^3 + a_2x^2 + a_1x + a_0 \qquad (2\text{-}1)$$

where a_3, a_2, a_1, and a_0 are coefficients. This might be evaluated at any point x by finding the various powers of x, multiplying them by the appropriate coefficients and adding all the terms. All this computation must be performed for each x desired. For path control x is usually constantly changing so that $y(x)$ must be constantly reevaluated. For real time operation such a method of polynomial evaluation requires a large amount of fast computing equipment. The economic requirements become prohibitive.

Another method of polynomial generation involves the use of connected integrators. For a third order polynomial the third derivative is constant. Integration of the third derivative produces the second derivative which in turn when integrated produces the first derivative. Integration of the first derivative yields the function. Thus, integrators can be cascaded as shown in Fig. 2.2 to generate the desired function. Appropriate initial values of the derivatives and of the function must, of course, be set in. Now as x increases the polynomial y(x) is continuously generated. Such a technique for function generation has been used previously employing mechanical integrators.[13,14] It is desired to achieve the same results using digital techniques. So integrations must be performed digitally.

Digital Integration

For the general case there exists no technique for achieving exact integration using digital techniques. However, there are several approximate numerical methods. Considering the amount and operational speed of the equipment needed, the method most suitable for use in real time path control applications is the summation of rectangular areas approximation. To see how this works, consider that it is desired to integrate the function y(x)

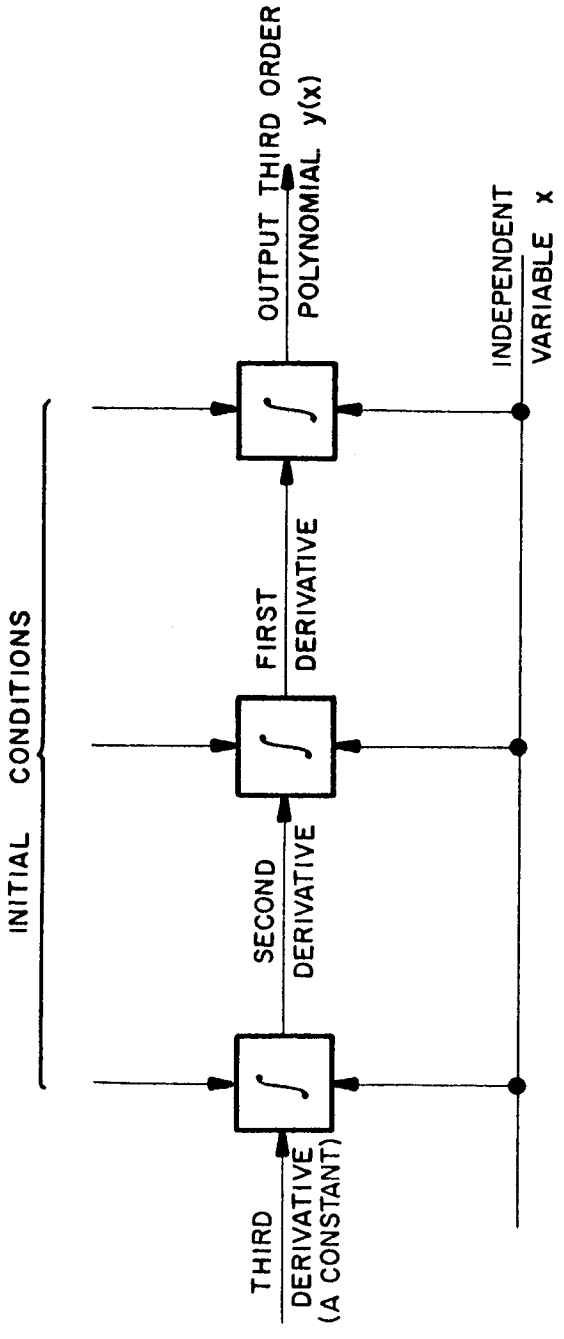


FIGURE 2.2 POLYNOMIAL GENERATOR

from x_0 to x_1 as shown in Fig. 2.3. The continuous integration is approximated by a finite summation as

$$\int_{x_0}^{x_1} y(x)\,dx \tilde{=} \sum_{i=0}^{N-1} y(x_i)\,\Delta x \qquad (2\text{-}2)$$

where $\Delta x = \frac{x_1 - x_0}{N}$, $x_i = x_0 + i\Delta x$, and N is the number of rectangles used in the approximation. The approximation becomes better as N is increased or correspondingly as Δx is decreased. Notice that since Δx is a constant

$$\sum_{i=0}^{N-1} y(x_i)\,\Delta x = \Delta x \sum_{i=0}^{N-1} y(x_i) \qquad (2\text{-}3)$$

So only the values of the $y(x_i)$ need be summed. The Δx appears as a scale factor term.

To see how the integration might be implemented, consider the arrangement of Fig. 2.4. Here, there are two digital registers and connecting arithmetic circuitry. The current value of y(x) corresponding to the current value of x is contained in the integrand register. As the numerical approximation to the integral, it is desired to have the running sum of the various $y(x_i)$ in the integral register. This is accomplished by adding the integrand register to the integral register every time a Δx command pulse occurs, i.e. every time the variable x increases

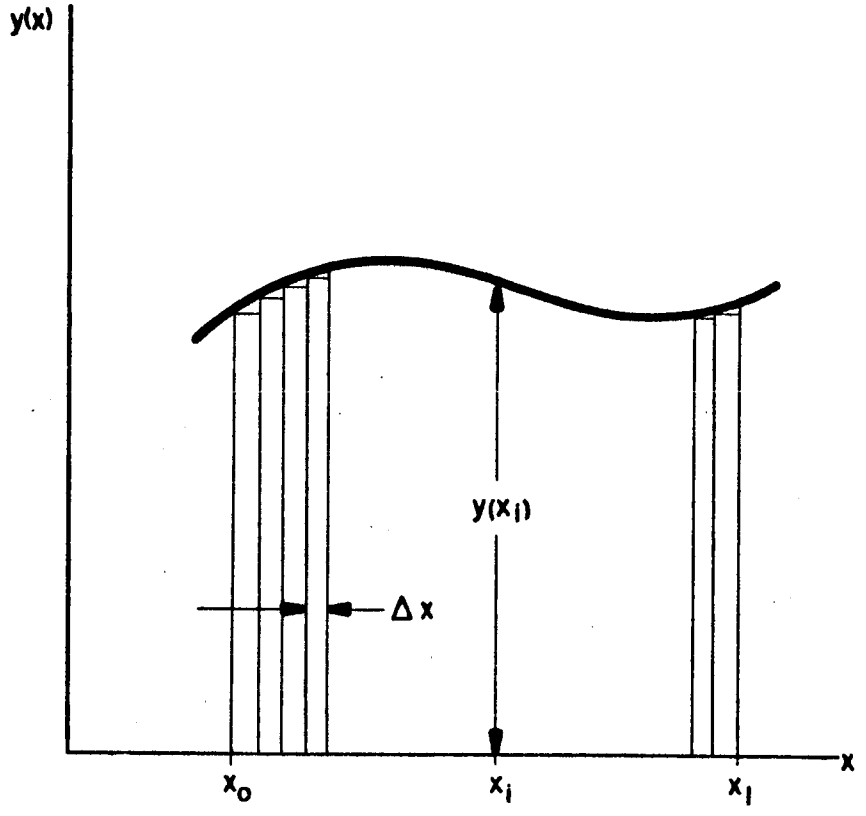


FIGURE 2.3 NUMERICAL INTEGRATION

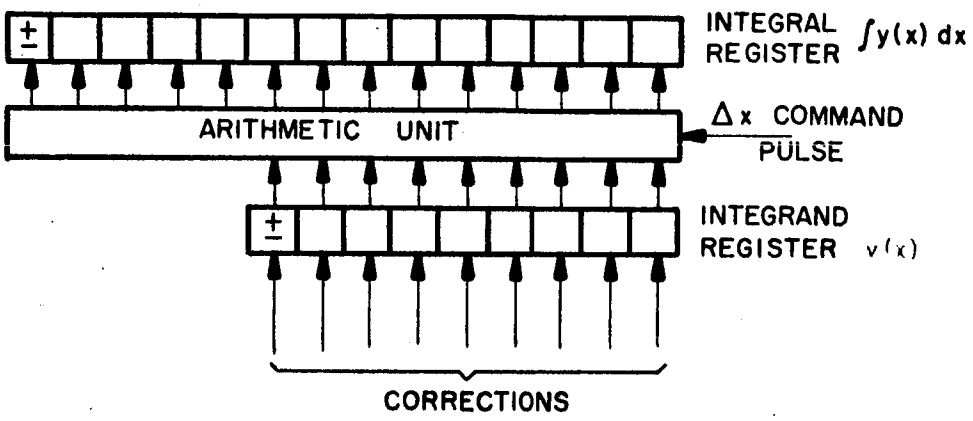


FIGURE 2.4 DIGITAL INTEGRATOR

by the increment Δx. The current value of x can be maintained by counting the Δx pulses. Notice that the integration is time independent and depends only on the arrival of Δx pulses. The value of $y(x)$ is maintained by adding appropriate corrections to the integrand register between arrival of Δx pulses.

Since this is a digital system, numbers in the registers are contained in the form of a sign bit and a binary representation of the magnitude. A natural binary representation will be used for positive numbers. However, for ease of operation of the arithmetic circuit, a special code will be used for the magnitude of negative numbers. This will be discussed later. At the present let it be assumed that the arithmetic circuitry can handle the numbers to preserve the proper sign and magnitude representations.

The scaling rule for the integrator registers follows directly from Eq. 2-2. This rule is that the weight of a bit in the integral register is equal to the product of the weight of the increment Δx and the weight of the corresponding bit in the integrand register.

The lengths of the registers shown in Fig. 2.4 have been arbitrarily set. For an actual integrator, these lengths are quite important because they govern the accuracy and capacity of the

integrator. The discussion of integrator accuracy and capacity will be postponed, however, until the entire path generator is designed and these considerations can be discussed for the total system.

Generation of a Third Order Polynomial Using Digital Integrators

Digital integrators can be connected in the manner that has previously been shown schematically in Fig. 2.2. This connection to generate a third order polynomial is shown in Fig. 2.5. Notice that a portion of the integral register of one integrator unit is used as the integrand register to the next unit. Also notice that since the third derivative of such a polynomial is constant, no updating is needed in that register. The maintenance of new values in the remaining registers is achieved through the action of the connected integrators. Initial conditions are set in the registers before starting the generation.

The integrators may be processed in either a parallel or serial fashion. Processing in a parallel fashion can introduce timing problems. To avoid these, serial processing has been chosen. The processing of the integrator units is done in the order 1, 2, 3. This leads to the following equations:

$$y(x_{i+1}) = y(x_i) + y'(x_i)\,\Delta x \qquad (2\text{-}4)$$

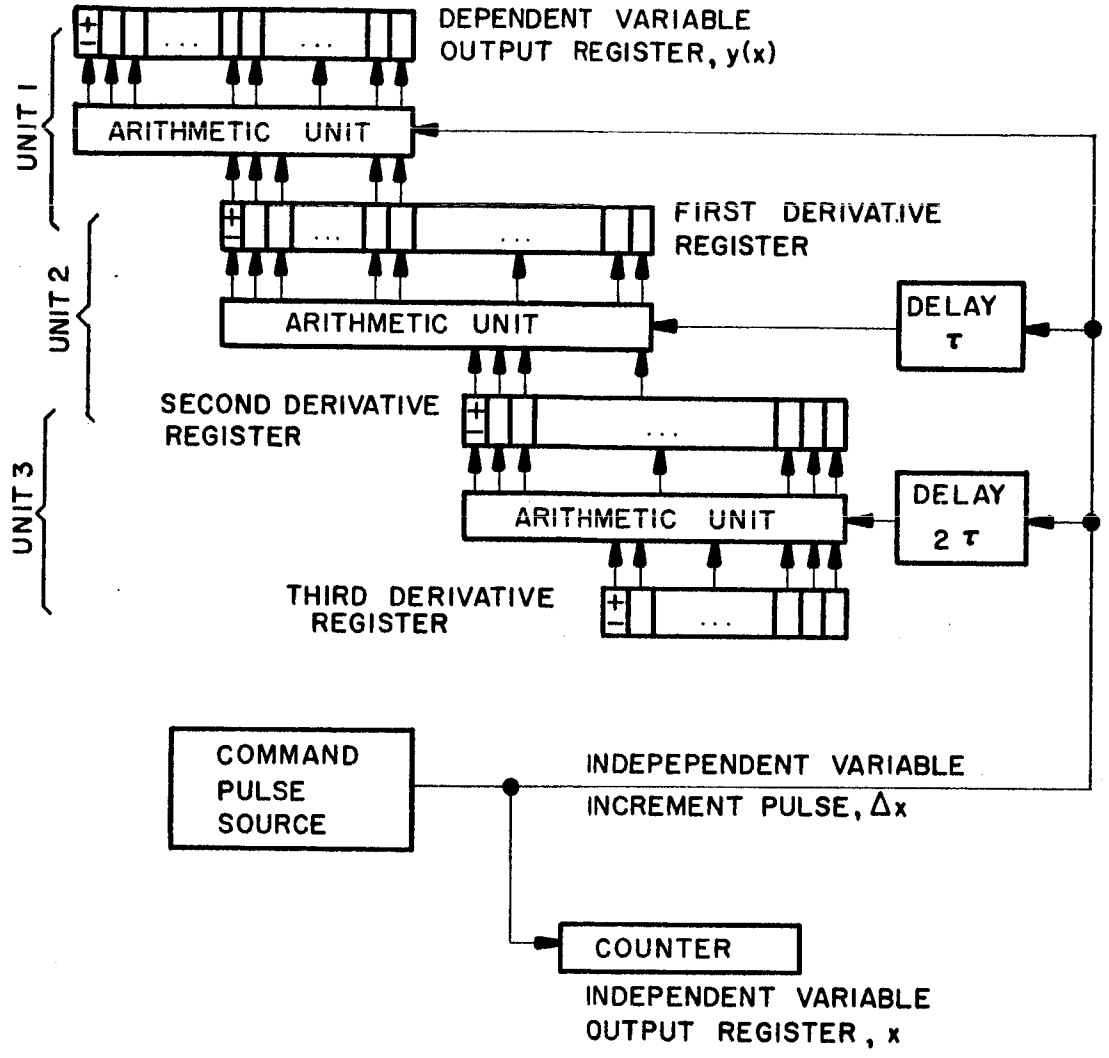


FIGURE 2.5 CONNECTED DIGITAL INTEGRATORS

$$y'(x_{i+1}) = y'(x_i) + y''(x_i)\,\Delta x \qquad (2\text{-}5)$$

$$y''(x_{i+1}) = y''(x_i) + y'''(x_i)\,\Delta x \qquad (2\text{-}6)$$

These are the truncated Taylor's series approximations and correspond to the appropriate rectangular integration formulas. Any other processing order does not lead to valid approximation formulas.

The proper processing order is represented in Fig. 2.5 by routing the Δx command pulses to the arithmetic units through delay elements. The Δx pulses are accumulated in a counter to maintain the current value of the variable x.

If appropriate initial conditions are introduced, the scheme of Fig. 2.5 can generate any single third order polynomial. However, the path is to be composed of interlaced polynomials. The problem then is to change from one polynomial to the next. As will be shown, this can be done by making corrections to two of the integrators. These corrections will involve only a function of the fourth differences of the dependent variable. Thus, with the addition of correction making circuitry, the system of Fig. 2.5 can then generate the entire desired path.

Integrator Corrections

Before the corrections needed in the integrators at the transition from one interval to another can be investigated, a general interpolation formula which can be used to represent the third order polynomial in any central interval is needed. Since it will be shown that corrections can be represented in the form of differences of the dependent variable, it is easier to deal with a general interpolation formula in terms of differences. Such an interpolation formula is developed in Appendix I.

To determine the necessary changes, consider the problem of the transfer from the generation of the interval between (x_1, y_1) and (x_2, y_2) of Fig. 2.1 to the interval between (x_2, y_2) and (x_3, y_3). For interpolation in the interval from $x = x_1$ to $x = x_2 = x_1 + h$, the function $y(x)$ is given, as shown in Eq. A.1-7 of Appendix I, by

$$y(x_1 + \alpha h) = y_1 + \alpha \Delta y_1 + \frac{\alpha(\alpha - 1)}{2} \Delta^2 y_0 + \frac{(\alpha + 1)(\alpha)(\alpha - 1)}{6} \Delta^3 y_0 \qquad (2\text{-}7)$$

where $0 \leq \alpha \leq 1$. The differences are defined in Table 2.1. For interpolation in the interval from $x = x_2$ to $x = x_3 = x_2 + h$, by a change of subscripts the appropriate interpolation function

TABLE 2.1

DIFFERENCE TABLE

x	y	Δy	$\Delta^2 y$	$\Delta^3 y$	$\Delta^4 y$
x_0	y_0				
		Δy_0			
x_1	y_1		$\Delta^2 y_0$		
		Δy_1		$\Delta^3 y_0$	
x_2	y_2		$\Delta^2 y_1$		$\Delta^4 y_0$
		Δy_2		$\Delta^3 y_1$	
x_3	y_3		$\Delta^2 y_2$		$\Delta^4 y_1$
		Δy_3		$\Delta^3 y_2$	
x_4	y_4		$\Delta^2 y_3$		$\Delta^4 y_2$
		Δy_4		$\Delta^3 y_3$	
x_5	y_5		$\Delta^2 y_4$		
		Δy_5			
x_6	y_6				

General Relationship

$$\Delta^m y_k = \Delta^{m-1} y_{k+1} - \Delta^{m-1} y_k \quad , \quad m = 1, 2, 3, \ldots$$

where $\Delta^0 y_k \equiv y_k$ and $\Delta^1 y_k \equiv \Delta y_k$

y(x) is then given by

$$y(x_2 + a\,h) = y_2 + a\,\Delta y_2 + \frac{a(a-1)}{2}\,\Delta^2 y_1 + \frac{(a+1)(a)(a-1)}{6}\,\Delta^3 y_1 \tag{2-8}$$

Differentiating Eq. 2-7 successively with respect to a yields

$$hy'(x_1 + a\,h) = \Delta y_1 + \frac{(2a-1)}{2}\,\Delta^2 y_0 + \frac{(3a^2-1)}{6}\,\Delta^3 y_0 \tag{2-9}$$

$$h^2 y''(x_1 + a\,h) = \Delta^2 y_0 + a\,\Delta^3 y_0 \tag{2-10}$$

$$h^3 y'''(x_1 + a\,h) = \Delta^3 y_0 \tag{2-11}$$

Letting a = 1 in this set of equations

$$hy'(x_2) = \Delta y_1 + \frac{1}{2}\,\Delta^2 y_0 + \frac{1}{3}\,\Delta^3 y_0 \tag{2-12}$$

$$h^2 y''(x_2) = \Delta^2 y_0 + \Delta^3 y_0 \tag{2-13}$$

$$h^3 y'''(x_2) = \Delta^3 y_0 \tag{2-14}$$

Successively differentiating Eq. 2-8 with respect to a gives

$$hy'(x_2 + a\,h) = \Delta y_2 + \frac{(2a-1)}{2}\,\Delta^2 y_1 + \frac{(3a^2-1)}{6}\,\Delta^3 y_1 \tag{2-15}$$

$$h^2 y''(x_2 + a\,h) = \Delta^2 y_1 + a\,\Delta^3 y_1 \tag{2-16}$$

$$h^3 y'''(x_2 + \alpha h) = \Delta^3 y_1 \tag{2-17}$$

Letting $\alpha = 0$ in the last three equations

$$hy'(x_2) = \Delta y_2 - \frac{1}{2}\Delta^2 y_1 - \frac{1}{6}\Delta^3 y_1 \tag{2-18}$$

$$h^2 y''(x_2) = \Delta^2 y_1 \tag{2-19}$$

$$h^3 y'''(x_2) = \Delta^3 y_1 \tag{2-20}$$

Subtracting Eq. 2-12 from Eq. 2-18 yields

$$\Delta(hy'(x_2)) = \Delta y_2 - \frac{1}{2}\Delta^2 y_1 - \frac{1}{6}\Delta^3 y_1 - \Delta y_1 \tag{2-21}$$

$$- \frac{1}{2}\Delta^2 y_0 - \frac{1}{3}\Delta^3 y_0$$

$$= \frac{1}{2}(\Delta^2 y_1 - \Delta^2 y_0) - \frac{1}{6}\Delta^3 y_1 - \frac{1}{3}\Delta^3 y_0$$

$$= (\frac{1}{2} - \frac{1}{3})\Delta^3 y_0 - \frac{1}{6}\Delta^3 y_1$$

$$= -\frac{1}{6}(\Delta^3 y_1 - \Delta^3 y_0)$$

$$= -\frac{1}{6}\Delta^4 y_0$$

Therefore, as x increases through x_2, a change in the first derivative of $-\frac{1}{6h}\Delta^4 y_0$ must be made in order to change from the polynomial appropriate for the interval x_1 to x_2 to that of the interval x_2 to x_3.

Subtracting Eq. 2-13 from Eq. 2-19 gives

$$\Delta(h^2 y''(x_2)) = \Delta^2 y_1 - \Delta^2 y_0 - \Delta^3 y_0 \qquad (2\text{-}22)$$

$$= \Delta^3 y_0 - \Delta^3 y_0 = 0$$

So, no change is necessary in the second derivative.

Subtracting Eq. 2-14 from Eq. 2-20 produces

$$\Delta(h^3 y'''(x_2)) = \Delta^3 y_1 - \Delta^3 y_0 \qquad (2\text{-}23)$$

$$= \Delta^4 y_0$$

Thus, a change of $\frac{\Delta^4 y_0}{h^3}$ must be made in the third derivative.

So once the integrators are initially set, in order to proceed through successive interpolation intervals, corrections to only two of the integrators must be made at the interval points. The two corrections are functions of only one piece of difference information, the fourth difference.[13,14]

Initial and Final Conditions

Initial conditions are needed in the integrator registers before starting the generator. Suppose that the path points are provided as shown in Fig. 2.1. The needed initial conditions will be found from the third order polynomial which passes through the first four points. The initial function values are the

starting coordinates x_0 and y_0. The remaining initial conditions are $y'(x_0)$, $y''(x_0)$, and $y'''(x_0)$. These are the derivatives of the polynomial evaluated at x_0. Since it is not desired to generate a different polynomial upon reaching (x_1, y_1), no correction is introduced here. Once point (x_2, y_2) is reached, the proper fourth difference exists and the generator proceeds in the normal fashion.

The equation, in terms of differences, of the function passing through the first four points was developed in Appendix I as Eq. A-1-6 and is

$$y(x) = y_0 + \frac{\Delta y_0}{h}(x - x_0) + \frac{\Delta^2 y_0}{2h^2}(x - x_0)(x - x_1) \qquad (2\text{-}24)$$

$$+ \frac{\Delta^3 y_0}{6h^3}(x - x_0)(x - x_1)(x - x_2)$$

The various derivatives are obtained as follows.

$$y'(x) = \frac{\Delta y_0}{h} + \frac{\Delta^2 y_0}{2h^2}\left((x - x_0) + (x - x_1)\right) \qquad (2\text{-}25)$$

$$+ \frac{\Delta^3 y_0}{6h^3}\left((x - x_0)(x - x_1) + (x - x_0)(x - x_2)\right.$$

$$\left. + (x - x_1)(x - x_2)\right)$$

$$y''(x) = \frac{\Delta^2 y_0}{h^2} + \frac{\Delta^3 y_0}{3h^3} \left((x - x_0) + (x - x_1) + (x - x_2) \right) \quad (2\text{-}26)$$

$$y'''(x) = \frac{\Delta^3 y_0}{h^3} \quad (2\text{-}27)$$

Evaluating these derivatives at $x = x_0$ gives

$$y'(x_0) = \frac{\Delta y_0}{h} + \frac{\Delta^2 y_0}{2h^2} (-h) + \frac{\Delta^3 y_0}{6h^3} (-h)(-2h)$$

$$= \frac{1}{h} \left(\Delta y_0 - \frac{\Delta^2 y_0}{2} + \frac{\Delta^3 y_0}{3} \right) \quad (2\text{-}28)$$

$$y''(x_0) = \frac{\Delta^2 y_0}{h^2} + \frac{\Delta^3 y_0}{3h^3} \left((-h) + (-2h) \right)$$

$$= \frac{1}{h^2} \left(\Delta^2 y_0 - \Delta^3 y_0 \right) \quad (2\text{-}29)$$

$$y'''(x_0) = \frac{\Delta^3 y_0}{h^3} \quad (2\text{-}30)$$

Thus, the initial conditions for the derivative registers have been evaluated in terms of the initial differences.

The last interval is generated by assuming that the last fourth difference is zero, i.e. no correction is given the integrators at the start of the last interval. Thus the function

generated over the last interval is a segment of the third order polynomial which passes through the last four points.

The system of Fig. 2.5 is modified as shown in Fig. 2.6 to handle initial conditions and corrections. The basic operation at each interval point is the addition of information from the input data source to the appropriate derivative registers. Initial clearing instructions are included if the input information consists of initial conditions. Otherwise, no clearing is done and the necessary corrections are added.

Use of this initial condition feature allows discontinuous paths, i.e. paths in which there is a sharp change in derivatives at a point, to be generated. This is done by setting in new initial conditions to the derivative registers at the point of discontinuity. The x and y values do not change at such a point so no provision is needed to enter new conditions here. Thus initial conditions are set in the output registers only at the start of any path. This is done manually. Changing these initial conditions allows various offsets to be introduced.

Representation of Numbers

In order that an integrator unit may handle all possible cases, both positive and negative numbers must be representable in the registers. This implies that each register must contain a

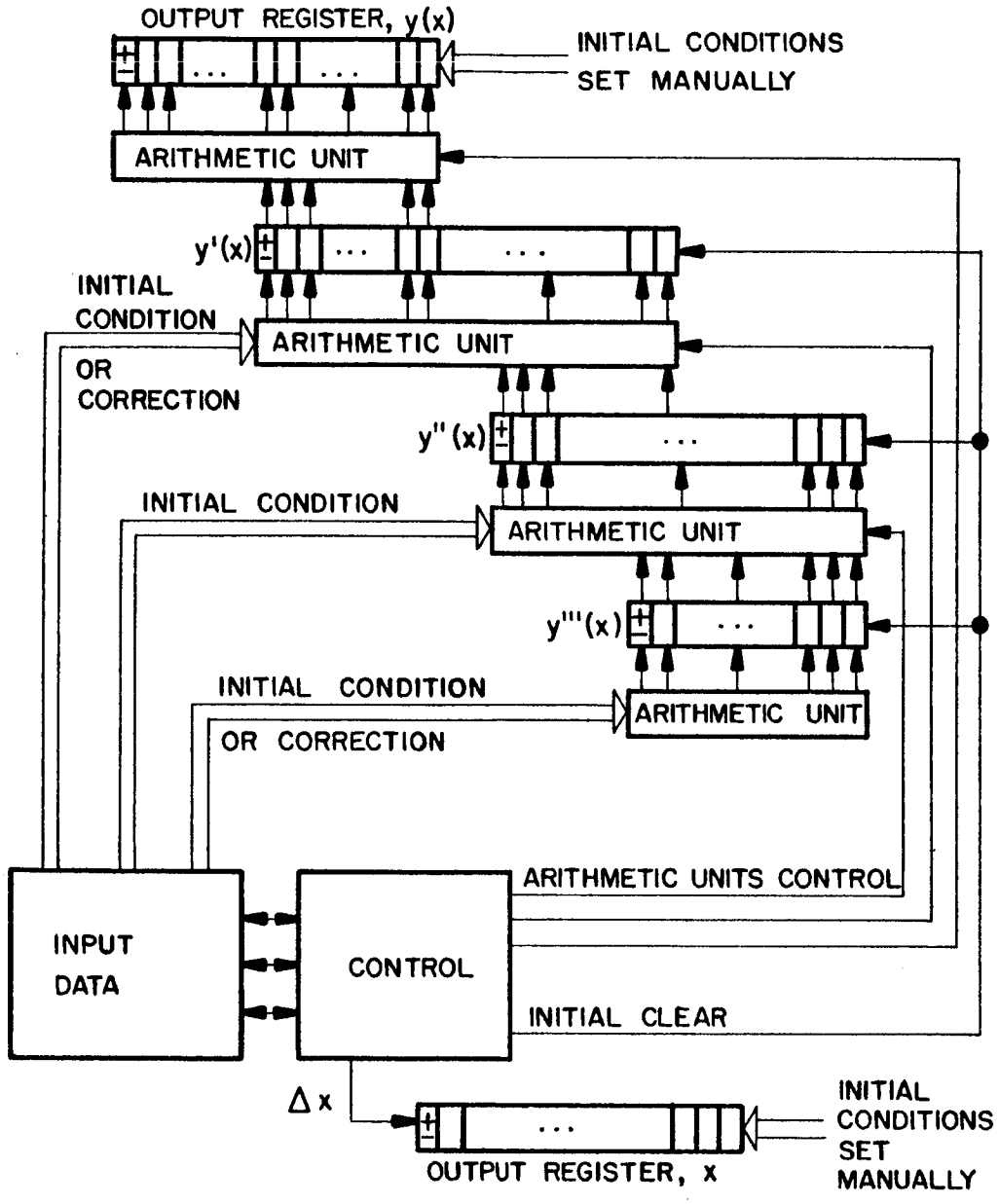


FIGURE 2.6 CORRECTION AND INITIAL CONDITION PROVISIONS

sign bit plus a representation of the magnitude of the number contained. The coding of the magnitude can be done in several ways.[3]

A natural binary representation can be used for the magnitude of both positive and negative numbers. If such a scheme is used, depending on the signs of the numbers in the registers, the basic summation operation of the integrator can be interpreted as either addition or subtraction. A rather complicated arithmetic unit with special sign circuitry then becomes necessary. If a one's complement representation is used for the magnitude of negative numbers, natural binary still being used for positive numbers, special correction and interpretation circuitry is still needed in the arithmetic unit. However, if two's complement is used to represent the magnitude of negative numbers, there is no need for extra sign or function determination circuitry. Only a straight binary adder is needed where the sign bits, a 0 for positive numbers and a 1 for negative numbers, are treated the same as the other bits. Overflow carrys from the last stage are neglected.

The addition of two positive numbers according to the rules of binary arithmetic then results in the correct natural binary result and is still positive. The addition of a positive number and

the two's complement representation of a negative number results in the correct natural binary answer if positive or in correct two's complement form if negative. The addition of two two's complement representations, i.e. two negative numbers, gives the correct two's complement answer.

To verify the last two statements, consider operations using the two positive numbers A and B. The numbers and any possible results are to be bounded in magnitude by 2^{n-1}, i.e. only n-1 register bits are required to represent the magnitudes of the numbers. The n^{th} register bit contains the sign. This bounding is done since overflows are to be neglected in the chosen scheme. Thus, scaling must be such that the results themselves do not cause any overflows.

The operation A-B is achieved by implementing A+(-B). The two's complement representation of (-B) considering the sign is (2^n-B). Physically, the two's complement of a binary number is formed by complementing every bit of the number, including the sign bit, and then adding 1, i.e. the one's complement, 2^n-1-B, is first formed and then a 1 is added to give 2^n-B, the two's complement. Then,

$$A + (-B) = A + (2^n - B) = 2^n + (A - B) \quad (2-31)$$

If $A \geq B$, there is an overflow 2^n and the positive number (A-B) results. If $A < B$, $A + (-B) = 2^n - (B - A)$ which is the correct two's complement representation.

Using proper two's complement representations, the operation (-A) + (-B) is performed as

$$(-A) + (-B) = (2^n - A) + (2^n - B) = 2^n + (2^n - (A + B)) \qquad (2\text{-}32)$$

Thus, an overflow 2^n results and the correct answer, a negative number, is in two's complement form.

Because of the described advantages, two's complement representation of negative numbers will be used in the integrator registers of the path generator. Since the feedback elements in the following servomechanisms are usually coded disks with representations that can be considered only in the positive range, the final outputs need only be positive numbers. Therefore, no conversion problems exist at the output. Input data, if negative, must be converted to a two's complement representation.

Arithmetic Unit

The arithmetic unit of an integrator must perform an accumulative summation, i.e. upon command the contents of the integrand register must be added to those of the integral register, the results appearing in the integral register. Since, through use

of two's complement representations, all operations have been reduced to binary addition, a simple ripple-carry binary adder as shown in Fig. 2.7 suffices. A parallel adder has been chosen to allow faster operation of the integrators. The logic symbols used here and in later diagrams are standard and are defined in the List of Symbols.

The adder is based upon transition coupled flip-flops, i.e. flip-flops which change state upon receipt of a level change in a given direction at a "T" input. The method of operation is as follows.

Upon command, the 1's of the integrand register are simultaneously gated into the "T" inputs of corresponding flip-flops in the integral register. Each flip-flop which receives an input changes state. A 1 to 0 transition means that a carry to the next stage is generated. Any carry is delayed, however, before it is gated to the "T" input of the next stage flip-flop. This delay prevents nearly simultaneous arrival of inputs to a flip-flop and insures reliable operation.

As can be seen, this type of adder operates upon carrys asynchronously. A finite settling time is needed before all transitions are completed and the correct answer appears in the integral register.

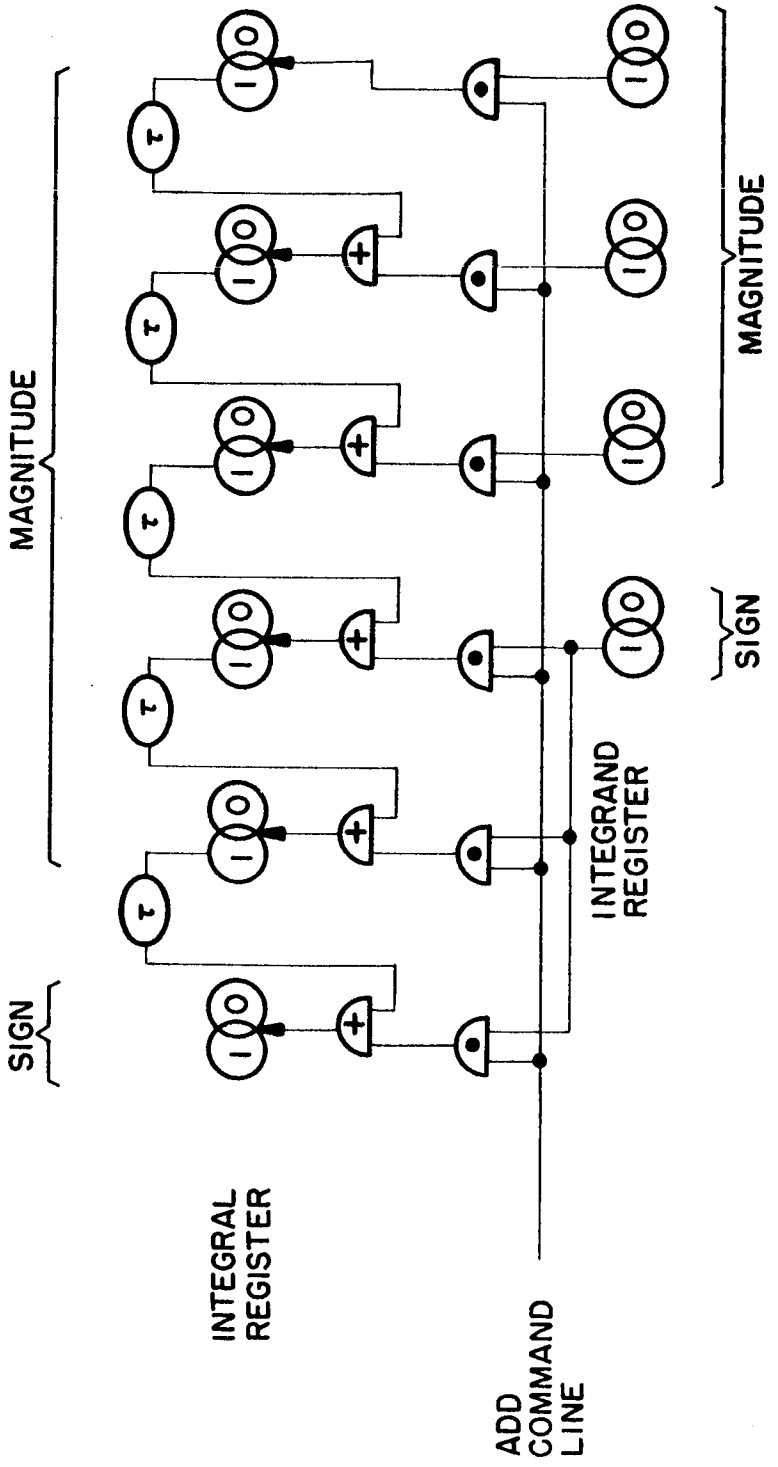


FIGURE 2.7 ARITHMETIC UNIT

To allow for the proper accumulation to take place, an integrator unit of the path generator has an integral register which is much longer than the integrand register. If the sign of the contents of the integrand register is positive, no special arrangement is needed to match its length with that of the integral register. If the sign of the integrand register is negative, however, in order to present the correct two's complement to the integral register, the length must be extended through use of the sign bit as shown in Fig. 2.7.

Special Features

In the examples of path generation in this chapter, x has always been the independent variable and y the dependent variable. In this, the normal mode of operation, the increment of the independent variable is added to the x output register and the first derivative register is added to the y output register. If this mode were fixed, however, an extremely large first derivative register would be required to generate all possible path slopes. To avoid this, so that a generator can be constructed using a modest amount of equipment, the decision is made that for a slope greater than 1 the roles of x and y will be interchanged. Thus, the maximum value that need be contained in the first derivative

register is 1. Special considerations must be taken in the input data preparation as a result of this restriction.

The implementation of the path generator incorporating the interchange feature is shown in Fig. 2.8. Here, the address of the independent variable is supplied by the input data source. This address controls the gating to the x and y registers and governs which receives inputs from the first derivative register and which receives the independent variable increment.

Under certain conditions, it may be desired to change the basic scale of the independent variable or to change the interpolation direction along this axis. Scaling changes are achieved by changing the weight of the added independent variable increment. Positive direction is achieved by adding the increment; negative direction, by adding the two's complement of the increment. The increment weighting and direction are also specified by the input data source.

Use of the initial condition, variable interchange, scaling, and interpolation direction features allows closed contours to be generated. An example of this is presented in Chapter IV.

Since paper tape will be used as the data source in the final constructed system, it is shown as such in Fig. 2.8. Control of the tape reader and integrators is represented only in

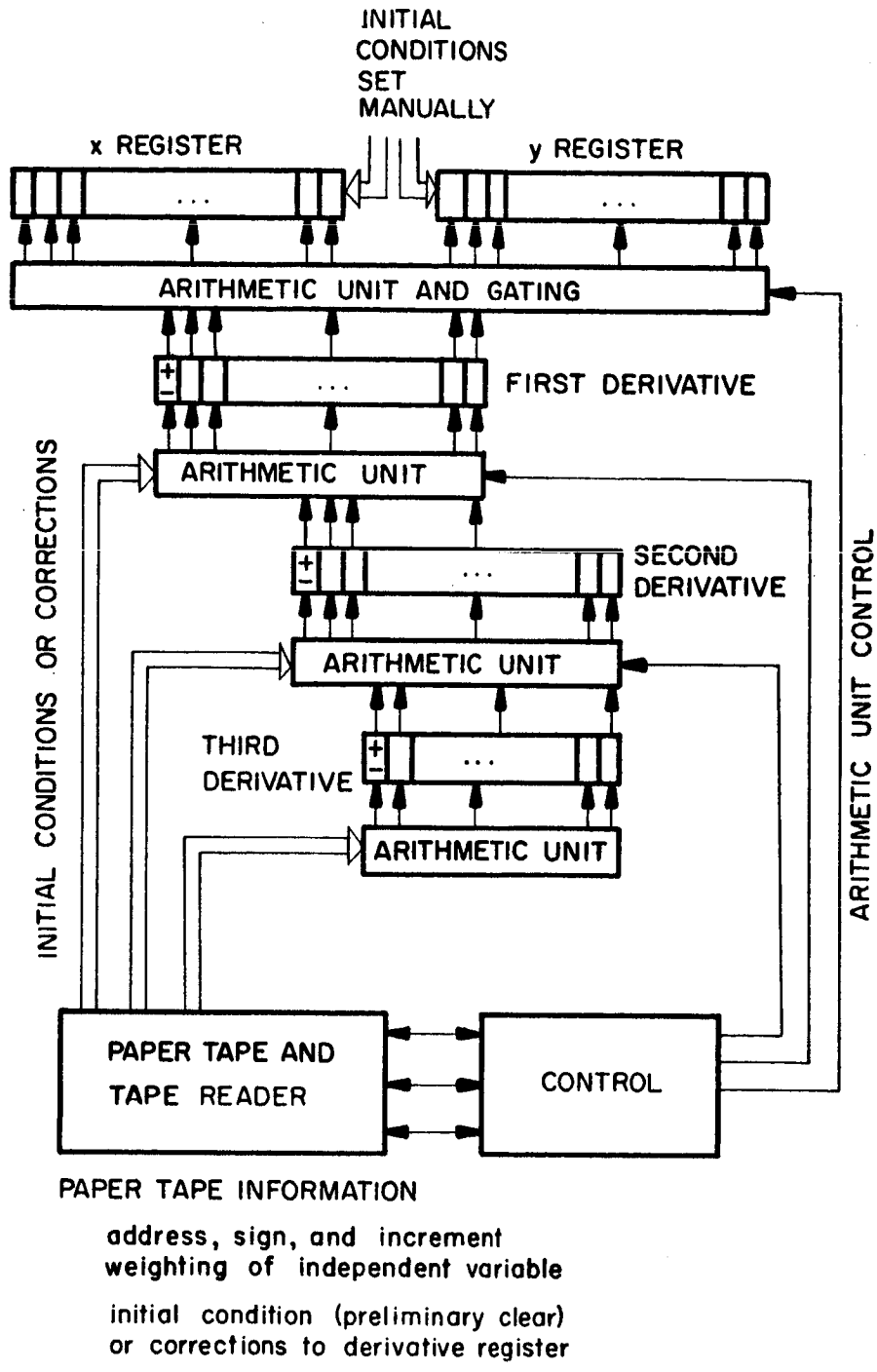


FIGURE 2.8 PATH GENERATOR WITH VARIABLE INTERCHANGE FEATURE

block diagram form here. Considerations of actual control circuit design will be postponed until the integrator registers are scaled.

CHAPTER III

SCALING, ERROR ANALYSIS, AND FINAL DESIGN OF THE PATH GENERATOR

In order to complete the design of the path generator, the weights of the most significant and least significant bits of each register and the normal weight of the increment of the independent variable must be determined. The basic interval between corrections must also be assigned. All scaling will be performed in units. The physical weighting of a unit is assigned by the following servomechanisms.

The weights of the most significant bits of the registers can be determined using equipment limitations and restrictions on the allowable functions which can be generated. An error analysis of the connected digital integrators must be made in order to determine the remaining weights.

First, for convenience, let the basic interpolation interval along the independent variable axis be $2^3 = 8$ units. Any other power of 2 could just as conveniently be chosen as the interval. A choice of other than a power of 2 as an interval introduces considerable hardware complications.

Determination of Weights of Most Significant Bits

The weight of the most significant bit of an output function register is chosen as 2^6 units to correspond with the most significant bit of the feedback encoder disks used in the following servomechanisms.

The limit has been set on the first derivative that its absolute value be ≤ 1. To correspond with this restriction, the most significant bit of the first derivative register is assigned a weight of 2^0. This weighting actually allows a maximum slope bounded by 2 units when contributions from the remaining bits are considered. Thus, for slight trespasses on the limit of a slope of 1 no switching of the x and y roles is necessary to continue generation of the function.

Using the limit set on the first derivative, let the limit be set on the second derivative in the following way. Consider that the second derivative is at its maximum value throughout an interval. This maximum value should be large enough to cause the first derivative to change from a maximum positive value to a maximum negative value or vice versa. That is,

$$\int_0^8 \left| y''(x) \right|_{max} dx = 2$$

or

$$\left| y''(x) \right|_{max} = \frac{2}{8} = 2^{-2} \qquad (3\text{-}1)$$

This is an absolute upper bound and can be approached using a register whose most significant bit has a weighting of 2^{-3}.

To set the upper bound on the third derivative, consider that it is at its maximum value throughout an interval. This should be large enough to cause the second derivative to change through half the possible range. That is,

$$\int_0^8 \left| y'''(x) \right|_{max} dx = 2^{-2}$$

or,

$$\left| y'''(x) \right|_{max} = 2^{-5} \qquad (3\text{-}2)$$

This again is an absolute upper bound and can be approximated using a register whose most significant bit has a weight of 2^{-6}.

All upper bounds have now been set on the integrator registers. These bounds limit the class of functions that can be produced by the path generator. However, since the bounds have been set quite conservatively, resulting problems should be at a minimum. An error analysis of the connected integrators must now be made in order to determine the remaining unknown weightings.

Error Analysis of Connected Digital Integrators

Integration by digital means is not an exact process.[2, 5, 12] Truncation error is introduced by the finite summation approximation of the continuous integration operation. The physical restriction of finite register lengths means that the input quantity must be rounded off or quantized. Thus, a roundoff error is introduced in the output. In the following, expressions for the truncation and roundoff errors in the output of a single digital integrator are first derived. These expressions are then used to analyze the three connected integrators of the path generator. In this analysis only the error in the output, or dependent, variable will be of concern. The independent variable, in this analysis x, can be maintained exactly by accumulating command increment pulses.

A single integrator is shown schematically in Fig. 3.1. The quantity f(x) is inserted in the integrand register. The quantity $\int f(x)\,dx$ with the appropriate initial conditions and limits is desired in the integral register. What is actually achieved is the summation $\sum_i f(x_i)\,\Delta x$, again with the appropriate initial conditions and limits. The difference between the integral and the rectangular approximation represents the truncation error, $E_t(N\Delta x)$.

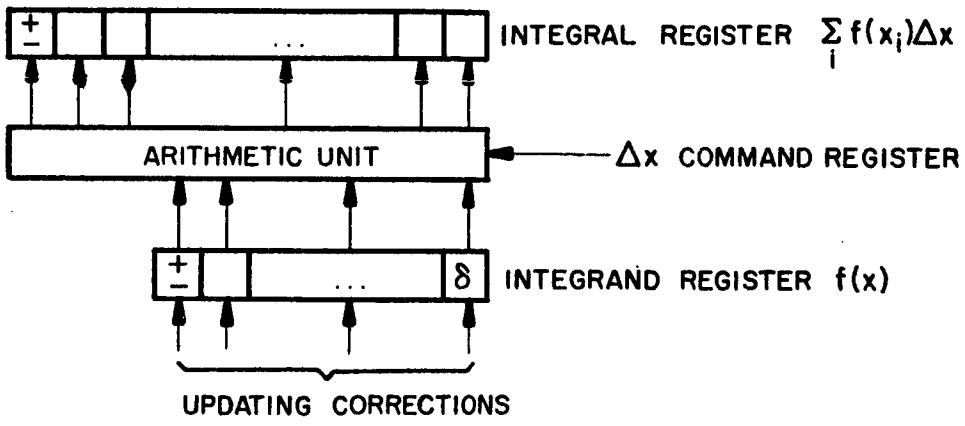


FIGURE 3.1 DIGITAL INTEGRATOR

$$E_t(N\Delta x) = \int_{x_0}^{x_1} f(x)\,dx - \sum_{i=0}^{N-1} f(x_i)\,\Delta x \qquad (3\text{-}3)$$

where $\Delta x = \frac{x_1 - x_0}{N}$, N is the number of rectangular segments used in the approximation, and $x_i = x_0 + i\Delta x$. Expressing the integral as the sum of integrals and moving the summation sign gives

$$E_t(N\Delta x) = \sum_{i=0}^{N-1} \left[\int_{x_i}^{x_i+\Delta x} f(x)\,dx - f(x_i)\,\Delta x \right] \qquad (3\text{-}4)$$

For $\int f(x)\,dx = F(x)$,

$$E_t(N\Delta x) = \sum_{i=0}^{N-1} \left[F(x_i + \Delta x) - F(x_i) - f(x_i)\,\Delta x \right] \qquad (3\text{-}5)$$

The Taylor's series expansion of $F(x_i + \Delta x)$ is

$$F(x_i + \Delta x) = F(x_i) + F'(x_i)\,\Delta x + \frac{F''(x_i + \beta\,\Delta x)\,(\Delta x)^2}{2\,!} \qquad (3\text{-}6)$$

where the last term is Lagrange's form of the remainder which results when the series is truncated after the term containing the first derivative and $0 \leq \beta \leq 1$. The parameter β must be determined for each different function.

Substituting Eq. 3-6 into Eq. 3-5 and recalling that $F'(x) = f(x)$, $F''(x) = f'(x)$, and $x_i = x_0 + i\Delta x$ yields

$$E_t(N\Delta x) = \sum_{i=0}^{N-1} \frac{f'(x_0 + i\Delta x + \beta\,\Delta x)\,(\Delta x)^2}{2\,!}$$

or,

$$E_t(N\Delta x) = \frac{(\Delta x)^2}{2} \sum_{i=0}^{N-1} f'(x_0 + i\Delta x + \beta\,\Delta x) \qquad (3\text{-}7)$$

Evaluation of this awaits knowledge of the nature of $f'(x)$.

If a number cannot be represented exactly within the finite register length of the integrand register, it must be rounded off. Due to the nature of the operation of the integrator, this is not a rounding off of the least significant representable bit to the nearest value but instead is a neglect of all bits less than the least significant bit. Thus the upper bound on the roundoff error in the integrand register is the weight of its least significant bit. The use of a longer register so that the bits that are dropped have a very small weighting reduces this error. However, since the contents of the integrand register are used many times, even slight errors accumulate and can have a significant effect on the output.

Without dealing with specific functions, the actual amount of roundoff error cannot be determined. An upper limit can be found, however. Let a weighting δ be assigned to the least significant bit of the integrand register. The error in the integrand will be less than δ at each step. Then $E_r(N\Delta x)$, the value of the roundoff error in the integral register after N steps, is

$$E_r(\Delta x) < \sum_{i=0}^{N-1} \delta \, \Delta x$$

or,

$$E_r(N\Delta x) < N\delta \, \Delta x \qquad (3\text{-}8)$$

The total output error, E, is the sum of the truncation error E_t and the roundoff error E_r, or

$$E = E_t + E_r \qquad (3\text{-}9)$$

Then,

$$E(N\Delta x) < \left[\frac{(\Delta x)^2}{2} \sum_{i=0}^{N-1} f'(x_0 + i\Delta x + \beta \Delta x) \right] + N\delta \, \Delta x \qquad (3\text{-}10)$$

This is the output error for only one integrator unit. If the output of one integrator serves as the input of another, then the error of the first integrator is reflected in the output of the second. Thus in the connected integrators of the path generator, errors can be compounded. This compounding of errors will now

be investigated using as a model the representation of the path generator shown in Fig. 3.2. Here, and throughout the remaining discussion on scaling, Δx represents the normal command increment of the independent variable.

Any error in the output of the third integrator unit, E_3, is reflected through the two other units. Thus, any slight error here can be compounded into a serious error in the output function. It is important, therefore, that the error E_3 be kept as small as possible. To do this, let the output roundoff error, E_{r3}, be zero. This means that any initial conditions or corrections to the third derivative register must be entered exactly.

As was shown in Eq. 2-30, any initial condition to the third derivative register is a third difference divided by h^3, where h has already been chosen as 2^{-3} units. Let the least significant bit of the third derivative register be given a weighting of 2^{-12} units. Then the third difference must be specified exactly by a binary number whose least significant bit has a weighting of $\frac{2^{-12}}{2^{-9}} = 2^{-3}$ units. This means that the restriction must be placed that the dependent variable of any given path point must be specified exactly by a binary number whose least significant bit represents 2^{-3} units. As a result of this restriction on the

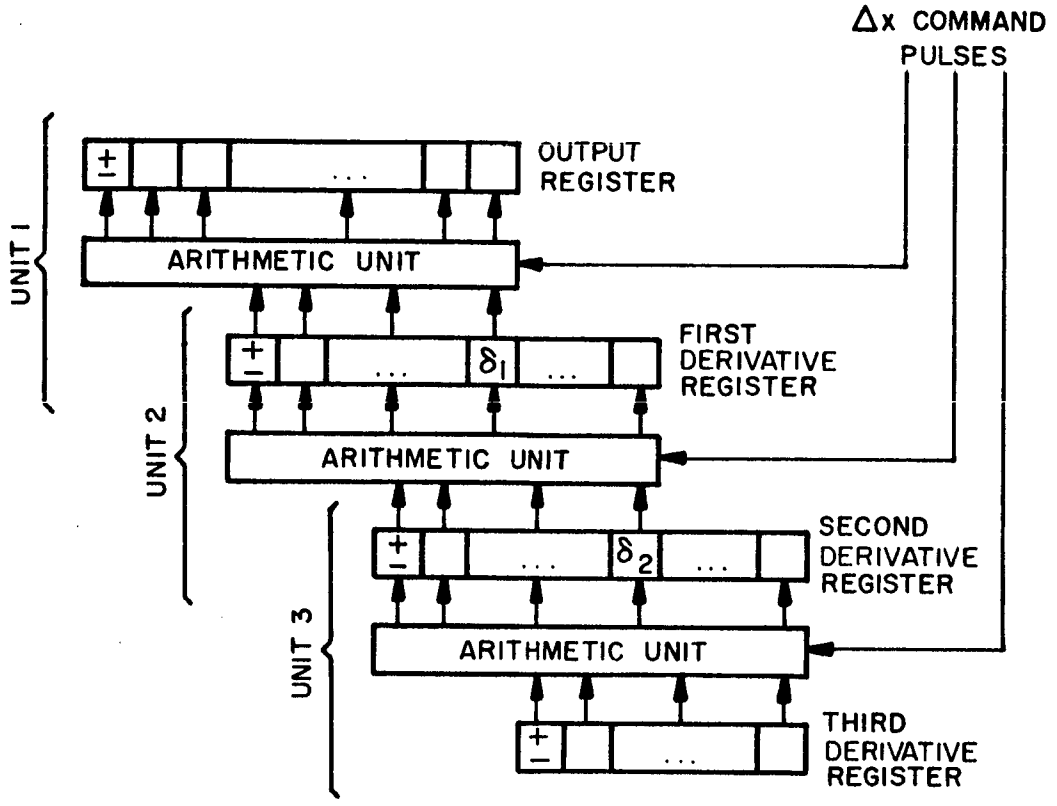


FIGURE 3.2 ERROR ANALYSIS MODEL OF PATH GENERATOR

dependent variable, any correction to the third derivative register which is a fourth difference divided by h^3 is also specified exactly to 2^{-3} units.

The figure of 2^{-12} units as the weighting of the least significant bit of the third derivative register has been chosen so that this register is eight bits long, seven magnitude bits and a sign bit. Inputs from one line of eight level paper tape can thus be easily made.

Since roundoff error has been eliminated, the output error E_3, which is the error in the second derivative, is then due only to the process of digital integration. Eq. 3-7 can be used to find this error. Notice that the third derivative is a constant for any third order polynomial being generated. Thus, the $f_3'(x_0 + i\Delta x + \beta\,\Delta x)$ to be used in Eq. 3-7 is $\frac{d}{dx}\,(y'''(x))\Big|_{x_o+i\Delta x+\beta\,\Delta x}$ equals the derivative of a constant and is therefore zero. The digital integration error is then also zero. Thus, the total output error is zero, i.e.

$$E_3(N\Delta x) = 0 \qquad (3\text{-}11)$$

The output error of the second integrator unit, which becomes the error in the first derivative, can now be found using Eq. 3-10.

Thus,

$$E_2(N\Delta x) < \left[\frac{(\Delta x)^2}{2} \sum_{i=0}^{N-1} f_2'(x_0 + i\Delta x + \beta\,\Delta x)\right] + N\delta_2\Delta x \quad (3\text{-}12)$$

where δ_2 is the weight of the least significant bit of the second derivative register which is used as an output to the first derivative register. For the second integrator unit, $f_2(x) = K_3 x$ where K_3 is the value of the third derivative and is a constant. Then,

$$f_2'(x_0 + i\Delta x + \beta\,\Delta x) = K_3 \quad (3\text{-}13)$$

Substituting this into Eq. 3-12 gives

$$E_2(N\Delta x) < \left[\frac{(\Delta x)^2}{2} \sum_{i=0}^{N-1} K_3\right] + N\delta_2\Delta x \quad (3\text{-}14)$$

or,

$$E_2(N\Delta x) < \frac{(\Delta x)^2}{2} NK_3 + N\delta_2\Delta x \quad (3\text{-}15)$$

The first integrator unit operates upon this error and introduces error of its own to give an error in the output dependent variable of

$$E_1(N\Delta x) < \left[\frac{(\Delta x)^2}{2} \sum_{i=0}^{N-1} f_1'(x_0 + i\Delta x + \beta\,\Delta x)\right] + N\delta_1\Delta x + \sum_{i=0}^{N-1} E_2(i)\,\Delta x \quad (3\text{-}16)$$

where δ_1 is the weight of the least significant bit of the first derivative register which is used as the output to a function register. Here

$$f_1'(x_0 + i\Delta x + \beta\,\Delta x) = K_2 + K_3(x_0 + i\Delta x + \beta\,\Delta x) \qquad (3\text{-}17)$$

where K_2 is the initial value of the second derivative. For a third order polynomial $\beta = \frac{1}{3}$. Then,

$$E_1(N\Delta x) < \left[\frac{(\Delta x)^2}{2} \sum_{i=0}^{N-1} [K_2 + K_3(x_0 + i\Delta x + \frac{\Delta x}{3})]\right] + N\delta_1\Delta x$$

$$+ \sum_{i=0}^{N-1} [\frac{(\Delta x)^2}{2} iK_3 + i\delta_2\Delta x]\ \Delta x \qquad (3\text{-}18)$$

Evaluating the summations gives

$$E_1(N\Delta x) < \frac{(\Delta x)^2}{2}\left[NK_2 + NK_3(x_0 + \frac{\Delta x}{3}) + \frac{N^2 - N}{2} K_3\Delta x\right] + N\delta_1\Delta x$$

$$+\frac{N^2 - N}{2} K_3 \frac{(\Delta x)^3}{2} + \frac{N^2 - N}{2}\delta_2 (\Delta x)^2 \qquad (3\text{-}19)$$

Collecting terms then gives the final error of the output of the first integrator unit as

$$E_1(N\Delta x) < (\Delta x)^2 N^2 (\frac{K_3\Delta x}{2} + \frac{\delta_2}{2}) \qquad (3\text{-}20)$$

$$+ \Delta x N (\delta_1 - \frac{\delta_2\Delta x}{2} + \frac{K_3\Delta x\, x_0}{2} - \frac{(\Delta x)^2 K_3}{3} + \frac{K_2\Delta x}{2})$$

This then is the error in the generated function. Setting a limit on this will allow δ_1, δ_2, and Δx to be determined.

Determination of Weights of Least Significant Bits

In order for Eq. 3-20 to be used in determining the remaining weights, a maximum allowable error limit must be set on the generated function. Let this limit be one unit for a third order polynomial generated over the maximum range of the independent variable. If a different limit is desired, similar procedures can be followed to achieve proper scaling.

First, let the error contributed by a part of Eq. 3-20 be less than or equal to $\frac{1}{2}$ unit, i.e. let

$$\frac{\delta_2}{2} (\Delta x)^2 N^2 + \delta_1 \Delta x N \leq \frac{1}{2} \tag{3-21}$$

For generation across the maximum interval, $N\Delta x = 2^7$, or

$$\frac{\delta_2}{2} (2^7)^2 + \delta_1 (2^7) \leq \frac{1}{2} \tag{3-22}$$

Eq. 3-22 is satisfied if

$$\delta_1 = 2^{-9} \tag{3-23}$$

and

$$\delta_2 = 2^{-15} \tag{3-24}$$

In order to stay well within the specified error limit, let the remaining portion of Eq. 3-20 be

$$(\Delta x)^2 N^2 \left(\frac{K_3 \Delta x}{2}\right) + \Delta x N \left(\frac{K_3 \Delta x\, x_0}{2} - \underline{\frac{\delta_2 \Delta x}{2} - \frac{(\Delta x)^2 K_3}{3}} + \frac{K_2 \Delta x}{2}\right) \leq \frac{1}{4} \qquad (3\text{-}25)$$

As will be seen, the two underlined terms are very small compared to the remaining terms in the last bracket and will be neglected. Regrouping the remaining terms of Eq. 3-25 then gives

$$\frac{K_3 (\Delta x)^2 N}{2} (\Delta x N + x_0) + \frac{K_2 (\Delta x)^2 N}{2} \leq \frac{1}{4} \qquad (3\text{-}26)$$

The bracketed term here has a maximum value of 2^7, the maximum range of the independent variable. Once K_2 and K_3 are determined, Eq. 3-26 can be solved for Δx.

The maximum values of K_2 and K_3 can be found considering that the slope of the generated path is bounded by an absolute value of 1. Thus, the maximum value of any change in slope is 2. This slope change is produced by the output of the second integrator unit, $\frac{K_3 x^2}{2} + K_2 x$. So to determine the maximum values, let

$$\frac{K_3 x^2}{2} + K_2 x \leq 2 \qquad (3\text{-}27)$$

Since x has a maximum value of 2^7,

$$\frac{K_3}{2}(2^7)^2 + K_2(2^7) \leq 2 \qquad (3\text{-}28)$$

If $K_2 = 0$, the maximum value of K_3 which solves this equation is

$$K_3 \Big|_{max} = 2^{-14} \qquad (3\text{-}29)$$

If $K_3 = 0$, the maximum value of K_2 which satisfies Eq. 3-28 is

$$K_2 \Big|_{max} = 2^{-6} \qquad (3\text{-}30)$$

Now using these values for K_2 and K_3, Eq. 3-26 becomes

$$\frac{2^{-14}}{2}(2^7)\,\Delta x\,(2^7) + \frac{2^{-6}}{2}\,\Delta x\,(2^7) \leq \frac{1}{4} \qquad (3\text{-}31)$$

or,

$$\frac{\Delta x}{2} + \Delta x \leq 2^{-2} \qquad (3\text{-}32)$$

or considering only a power of 2,

$$\Delta x \leq 2^{-3} \qquad (3\text{-}33)$$

Let Δx be 2^{-4}, then, since this satisfies Eq. 3-33. All register weights have now been determined.

Final Design

The weightings that have been determined and the scaling rule developed in Chapter II are used to obtain the final block

diagram for the path generator. See Fig. 3.3. The numbers included in the register bits represent the weighting of that bit in a power of 2. Notice that the registers of the two lower integrator units are lined up so that the weight of an integral register bit is equal to the weight of the corresponding integrand register bit times the weight of the independent variable increment, 2^{-4}. This same scaling rule is followed in the top integrator unit through use of internal circuitry in the arithmetic unit and gating block. The necessity of showing the interchange feature prevents the physical lining up of the block diagram representations of the output registers with the first derivative register. During operation, then, the 2^{0} bit of the first derivative register is added to the 2^{-4} bit of the dependent variable output register, etc. Under normal scaling, the independent variable increment is added to the 2^{-4} bit of the independent variable register. Recall that input information determines whether x or y is the independent variable.

In actual implementation, the registers will consist of flip-flops. Then, the connecting arithmetic units, shown here only as blocks, will be of the same type shown in Fig. 2.7.

Various control lines are shown in Fig. 3.3. For proper functioning of the path generator, these must be operated so that

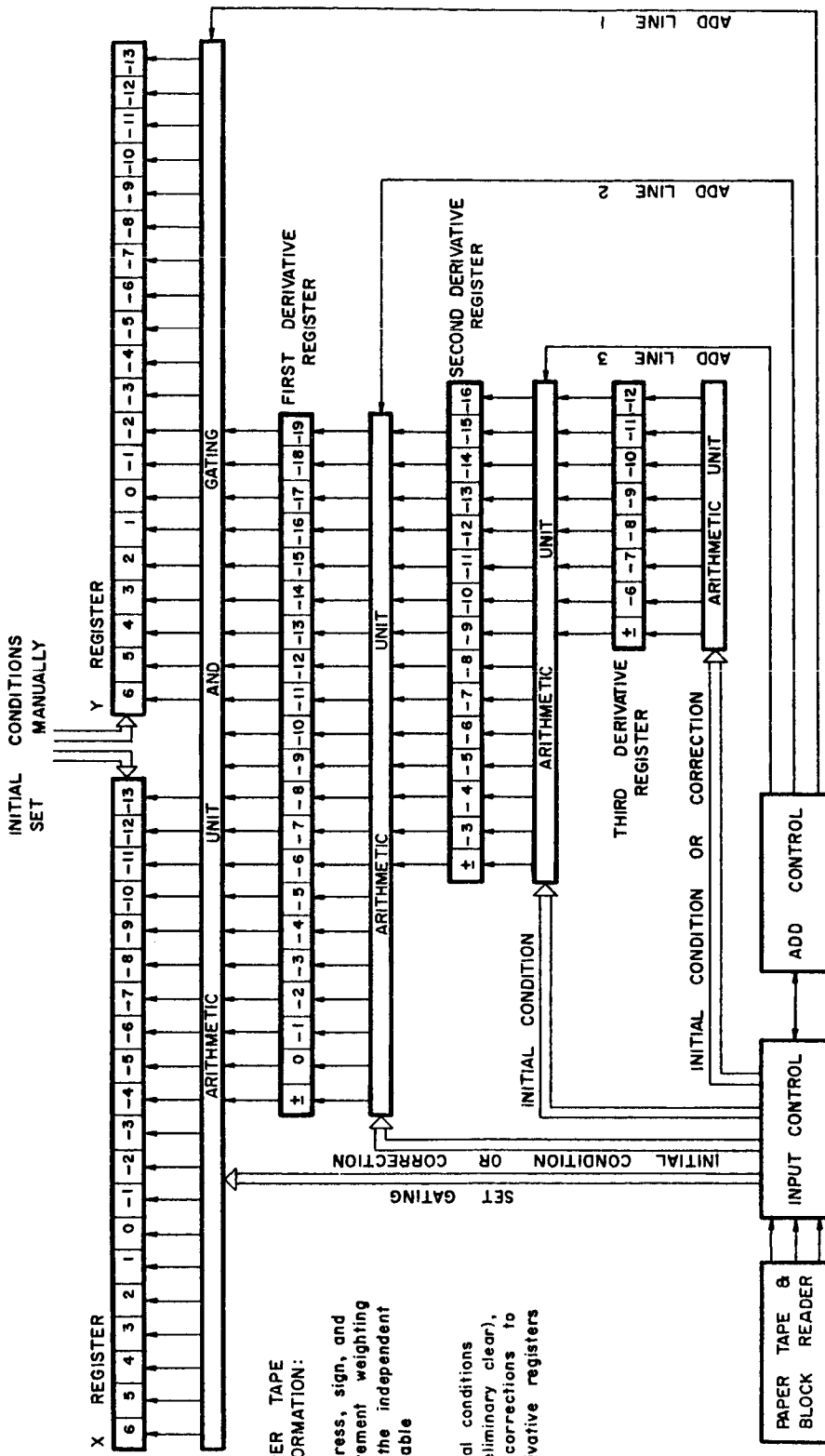


FIGURE 3.3 FINAL BLOCK DIAGRAM OF PATH GENERATOR

the control flow diagram of Fig. 3.4 is followed. This flow diagram implies the use of two gated counters, one to govern the cyclic additions of the integrators (Add Control), and one to provide tape reading control, setting of the gating, and control of inserting initial conditions or corrections (Input Control). For this reason, the control is represented in Fig. 3.3 by two blocks with the appropriate control lines emanating from the blocks.

The actual implementation of the path generator of Fig. 3.3 is described in Appendix II. The outputs of the path generator provide inputs to a two-dimensional plotter. This plotter, whose construction is described in Appendix III, represents the system to be controlled. Graphical output records from the combined systems are shown in the next chapter.

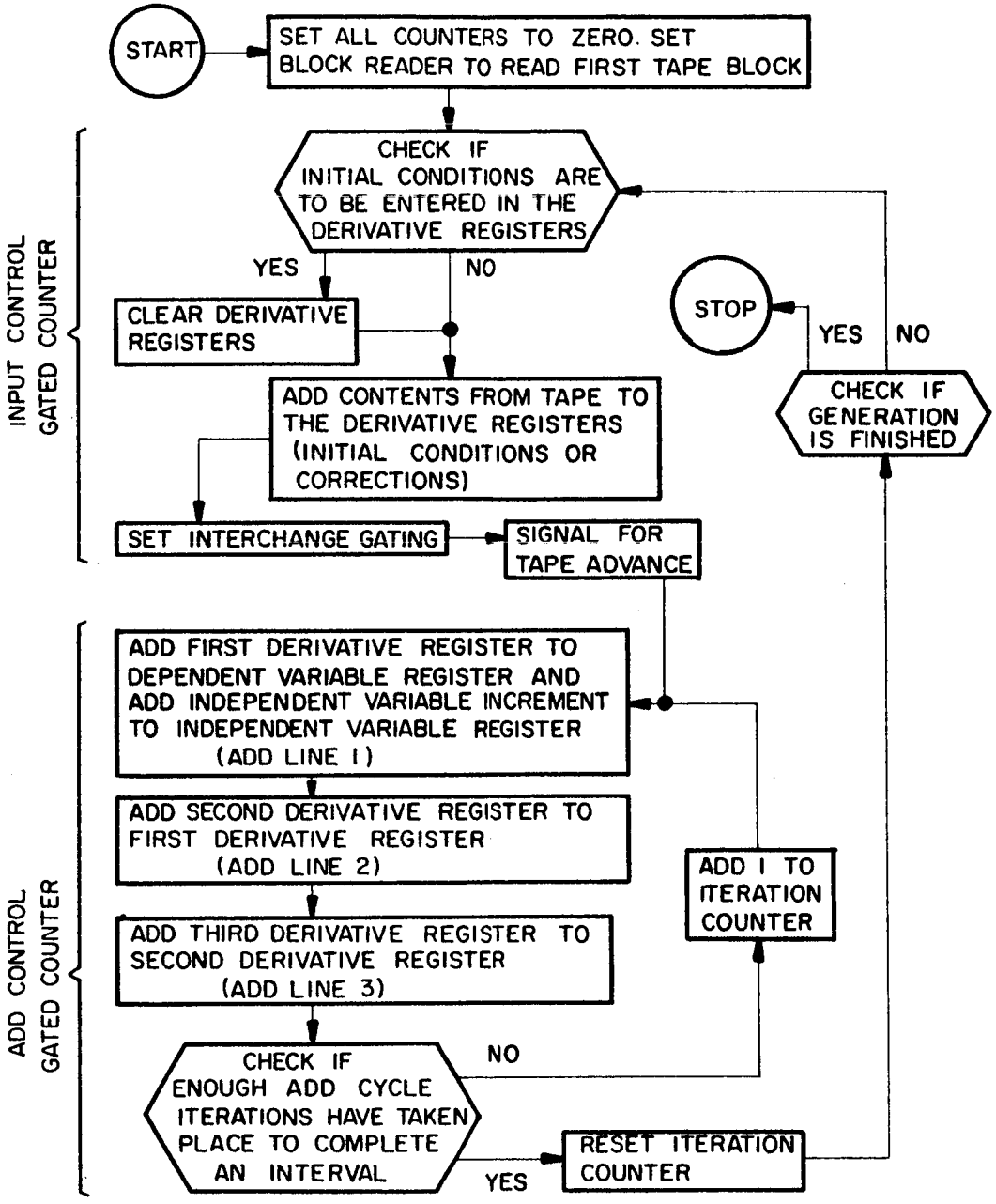


FIGURE 3.4 CONTROL FLOW DIAGRAM

CHAPTER IV

EXPERIMENTAL RESULTS

The path generator has been constructed using the Digital Synthesizer of the Case Digital Systems Laboratory. This Synthesizer features modular logic units which are connected using removable wired program boards.

Inputs to the generator are provided by a paper tape block reader. Each block of input tape contains the information to allow path generation over one interval of the independent variable. The use of a block reader eliminates the need for intermediate internal storage and simplifies the gating in of input information.

The outputs of the generator are used to control a two-dimensional plotting table. Each axis of the table contains an absolute digital servo drive. The plotter table produces a graphical record of the generated path.

A photograph of the combined elements is shown in Fig. 4.1. The Synthesizer is shown in the left background, the plotter table in the center, and the servo drive electronics and tape block reader at the right.

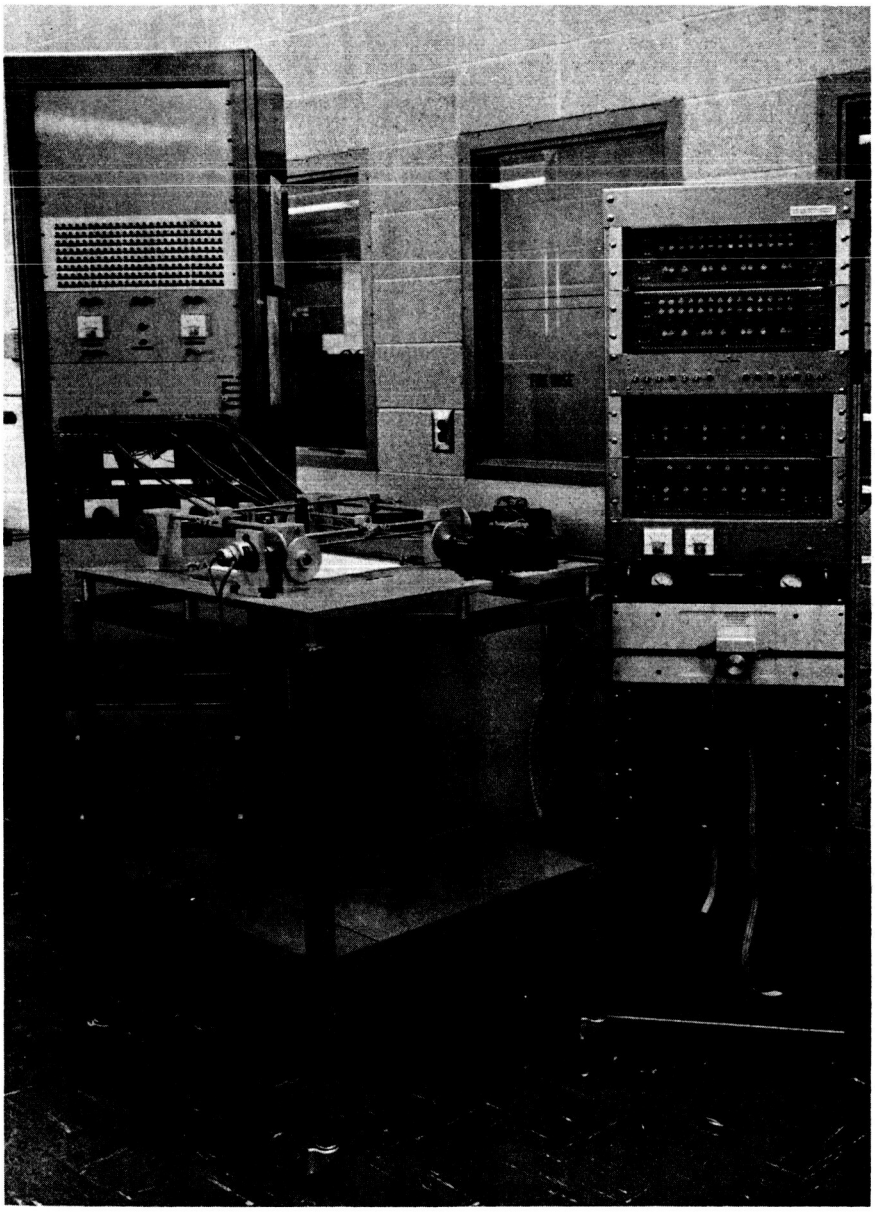


FIGURE 4.1 PHOTOGRAPH OF PATH CONTROL SYSTEM

It was pointed out in Chapter II that the path generator normally produces a single valued continuous path. However, modifications introduced by instructions on the input tape allow the use of the basic manner of operation to produce other types of paths. Thus, the clearing and setting in of new initial conditions in the derivative registers allows the generation of sharply discontinuous paths. Changing the direction of interpolation along the independent variable axis and/or the assignment of x or y as the independent variable allows various types of closed contours to be produced. The restriction on the magnitude of the first derivative means that a combination of these modifications must be used to generate certain paths.

To illustrate these features, two examples of path generation and the required data preparation will now be presented. The first will discuss a closed contour composed of various types of segments. The second will be concerned with the generation of a spiral.

Example 1

The path of this example is composed of various types of segments. Each segment is used to demonstrate a different type of path which can be generated. The path record produced by the

plotter table is shown in Fig. 4.2 along with the points through which the path should pass.

The section between points 1 and 9 demonstrates the possibility of linear interpolation in the generator for a slope magnitude less than 1 with respect to the x axis. The segment between points 9 and 16 shows the normal third order interpolation feature where +x is the independent variable. The section between points 16 and 23 presents straight line generation along the y axis. The portion between points 23 and 28 shows third order interpolation with +y as the independent variable. The section between points 28 and 39 shows straight line generation along the x axis. The last segment, between points 39 and 41, demonstrates linear interpolation for a slope magnitude greater than 1 with respect to the x axis. Further discussion of the path record will be given after the input data preparation is presented.

The coordinates of the path points along with the appropriate differences are tabulated in Fig. 4.3. The path points and differences are grouped according to path segments. For each path segment, the sign and address of the independent variable is given to the left of the grouping. The differences are then taken of the other variable. Using the data contained here, the

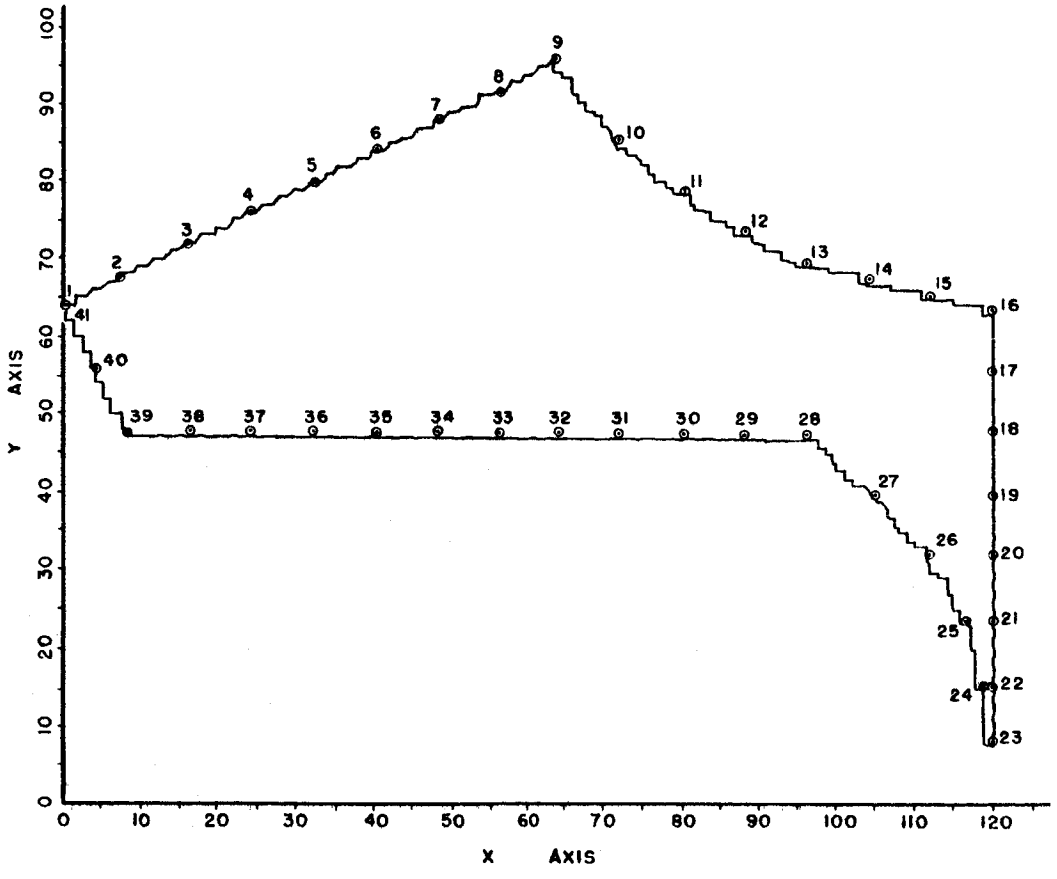


FIGURE 4.2 CLOSED CONTOUR

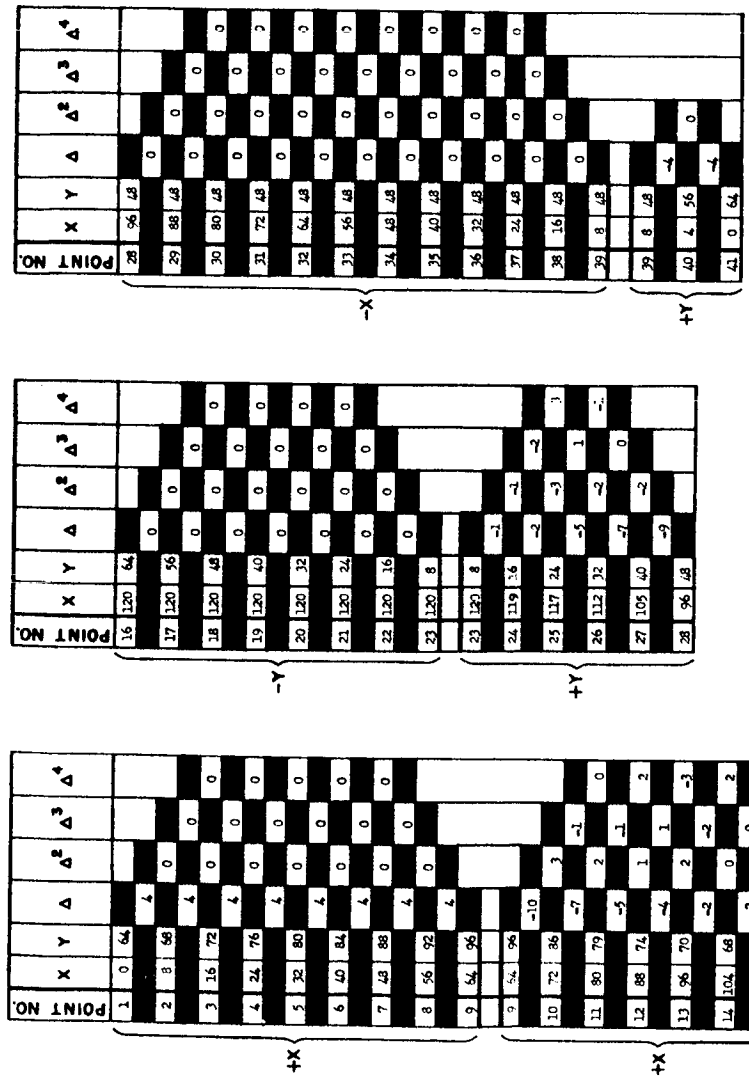


FIGURE 4.3 PATH POINTS AND DIFFERENCES FOR CLOSED CONTOUR

information to be entered on the control tape of the generator can be found.

Each tape block consists of seven lines. The information is entered on to the tape according to the format shown in Fig. 4.4. Thus, the first three lines provide information for the first derivative register. The next two contain information for the second derivative register. The sixth line provides inputs to the third derivative register. The seventh line gives gating control information, a stop instruction, and a preliminary clearing command when initial conditions instead of corrections are to be added to the derivative registers. The gating control information consists of the address and sign of the independent variable. Normal operation is with +x as the independent variable. Any changes to this must be entered. Holes are punched in the tape for 1 data bits and for commands. A 0 or the absence of a particular command or gating signal is represented by blank tape.

Initial conditions to the derivative registers are entered from tape at the start of the path and at the beginning of each new path segment. These are calculated according to Eqs. 2-28, 2-29 and 2-30. Examples of initial condition calculations and conversion to the proper tape coding are given in Appendix IV. Fourth difference corrections are entered in the first and third derivative

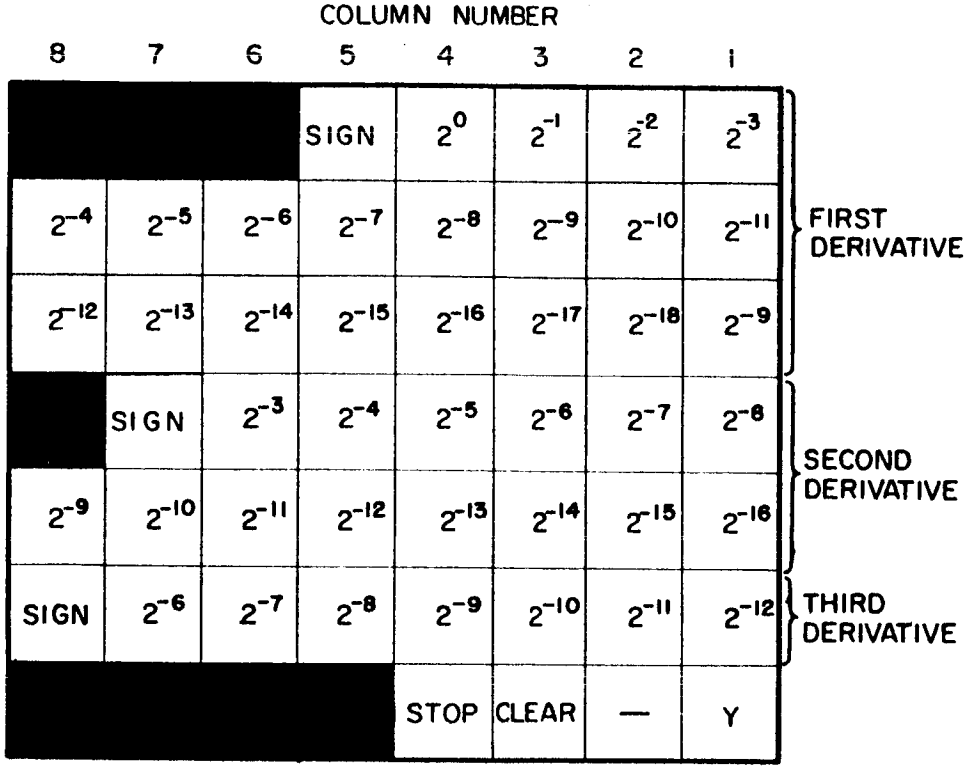


FIGURE 4.4 – TAPE FORMAT

registers at the other interval points. Initial conditions for the x and y registers at the start of the path are not obtained from the input tape but must be set in manually.

Since the derivative register initial conditions are calculated so that the path passes through the first four points of a segment, no correction is made at the second point of each new segment. Also since the final portion of each segment is to be the third order polynomial which passes through the last four points, no correction is made at the second to last point of each segment. For segments shorter than three intervals, i.e. segments which pass through less than four path points, lower than third order interpolation must be used. This condition is demonstrated better in Example 2.

Tables which allow the fourth difference corrections to be entered in the proper coding on the control tape are given in Appendix IV. Tables A. 4-1 and A. 4-2 give entries to the three lines of tape which provide inputs to the first derivative register. These tables give the proper coding for $-\frac{1}{6h}\Delta^4$ (dependent variable). Remember that a straight binary coding is used for positive numbers and a two's complement coding is used for negative numbers. The interval h was chosen as 2^3 units. Table A. 4-3 gives the coding for $\frac{1}{h^3}\Delta^4$ (dependent variable) which is the correction term to the third derivative register. Recall that

no corrections are needed for the second derivative register. Initial conditions must be introduced there, however, at the start of each new path segment.

Using the tabulated fourth differences, the coding tables, and the calculated initial conditions for each new interval, the information in the control tape which will allow the generation of the closed contour is then given in Fig. 4.5. Each tape block presents information which allows generation of the path over the next interval.

Returning to a discussion of Fig. 4.2, notice that due to the quantization effects of both the generated data and the plotting table and error in the path generator, the generated path does not always pass through the desired points exactly. However, there is never more than one unit error in any coordinate direction. The path does not close exactly, a one unit error remaining in the y direction at the end of the contour. This is due mainly to the quantization effect of the output of the generator. The final error determined by observing the total contents of the y output register is approximately 1/4 unit.

Example 2

The only feature not demonstrated by Example 1 is the use of variable interchange when the magnitude of the path slope

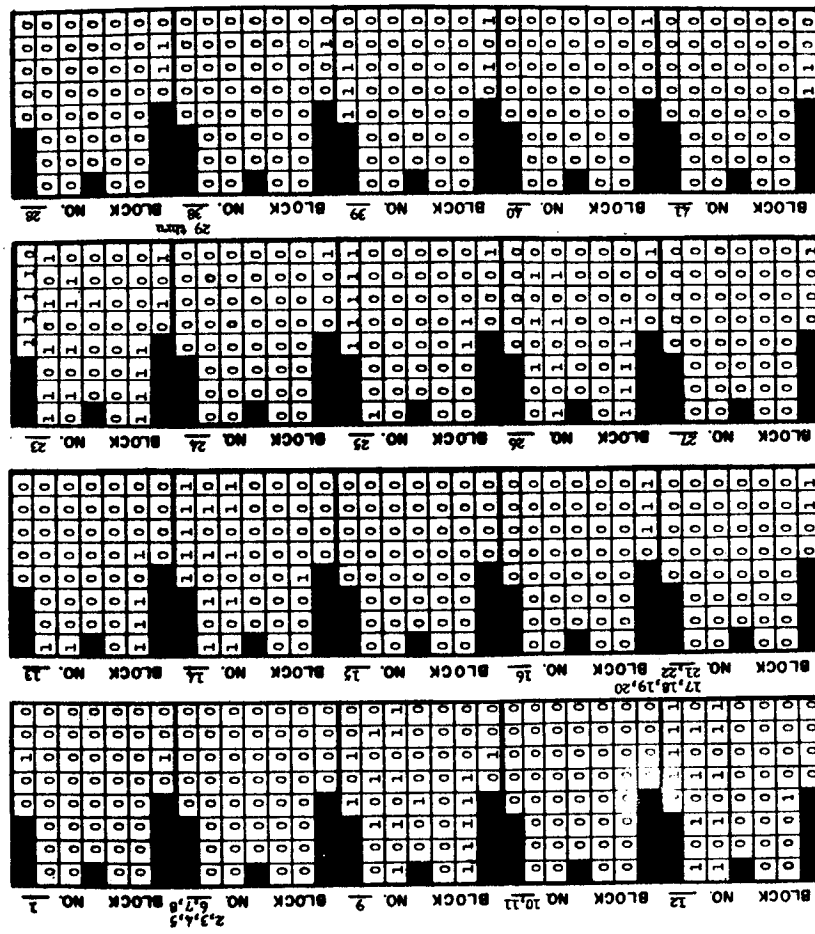


FIGURE 4.5 CONTROL TAPE INFORMATION FOR GENERATION OF CLOSED CONTOUR

changes through 1. Actually, the selection of interchange points can be made for slope magnitude values anywhere between 1/2 and 2 since the upper bound on the capacity of the first derivative register is 2. Thus the selection of interchange points is not too critical.

The interchange feature is now demonstrated through generation of a spiral. The output record along with the input path points is shown in Fig. 4.6. The independent variable interchange is made at points 14, 20, 23, 26 and 28. For the segments between points 1 and 14, 20 and 23, and 26 and 28, x is the independent variable. For the first and third of these segments, a positive interpolation directions is used. For the second section, interpolation takes place in the negative direction. For the remaining segments, y is the independent variable. A positive interpolation direction is used between points 14 and 20, and 28 and 30. A negative interpolation direction is used between points 23 and 26.

The coordinates of the path points along with the appropriate differences are tabulated in Fig. 4.7. For each path segment, the sign and address of the independent variable is again given to the left of the segment. The differences of the

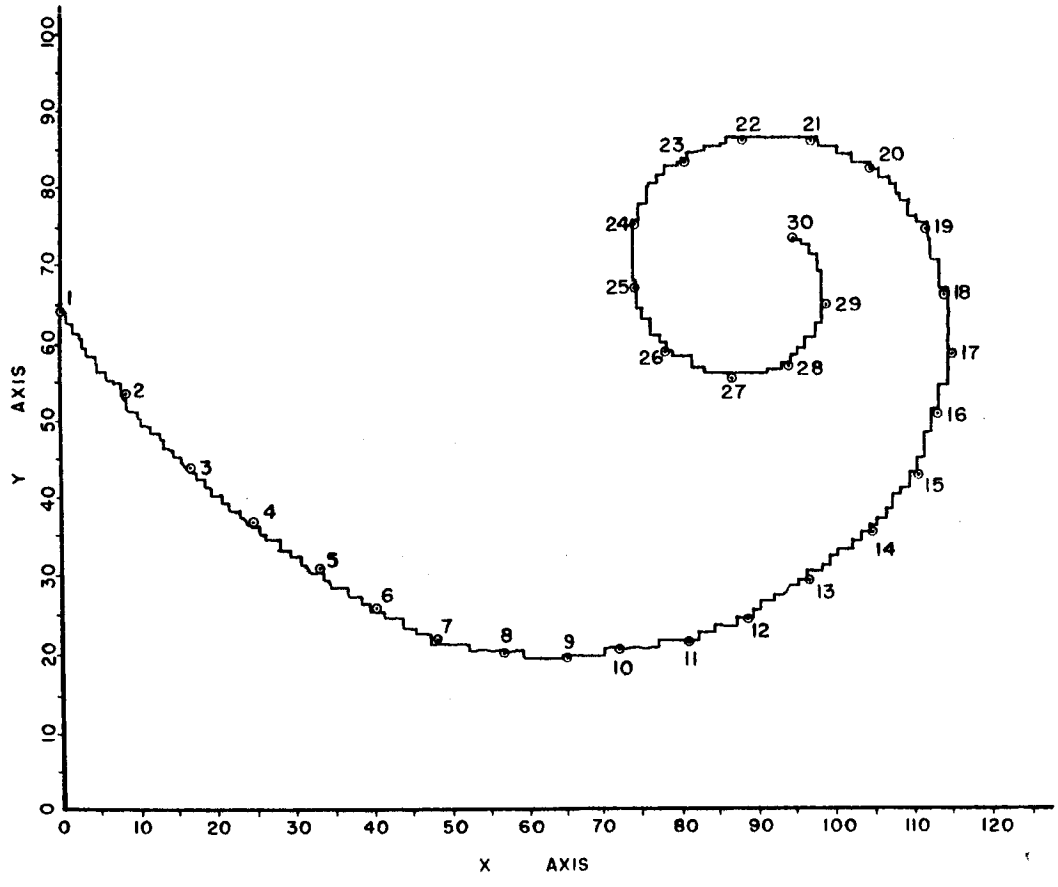


FIGURE 4.6 SPIRAL

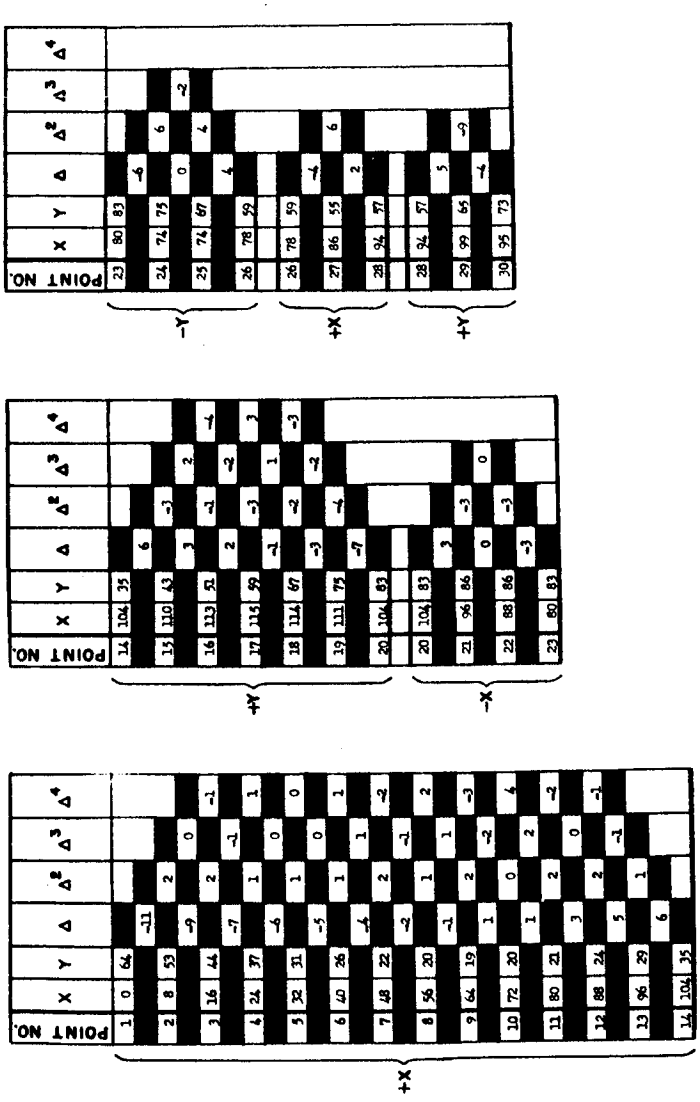


FIGURE 4.7 PATH POINTS AND DIFFERENCES FOR THE SPIRAL

other variable are then taken. The information to be entered in each tape block can be found using this data.

At each interchange point new initial conditions are introduced. Example of initial condition calculations are given in Appendix IV. The coding tables of Appendix IV are used in the same manner discussed in Example 1 to convert the fourth differences corrections to the proper tape coding.

The control tape information to allow generation of the spiral is then given as shown in Fig. 4.8. Again, no corrections are needed at the second point of a path segment or the next to last point of a segment. Notice that the last two segments only pass through three path points. Thus only second order interpolation is available here.

As in the previous example, due to quantization effects and error in the generator, the generated path of Fig. 4.6 does not pass through the desired points exactly. Notice, however, that there is never more than one unit error in a coordinate direction. Despite the many interchanges between x and y as the independent variable which must be made to produce this path, the generated path does pass through many of the path points including the final point with no error. This is a good indication of the accuracy of the path generator.

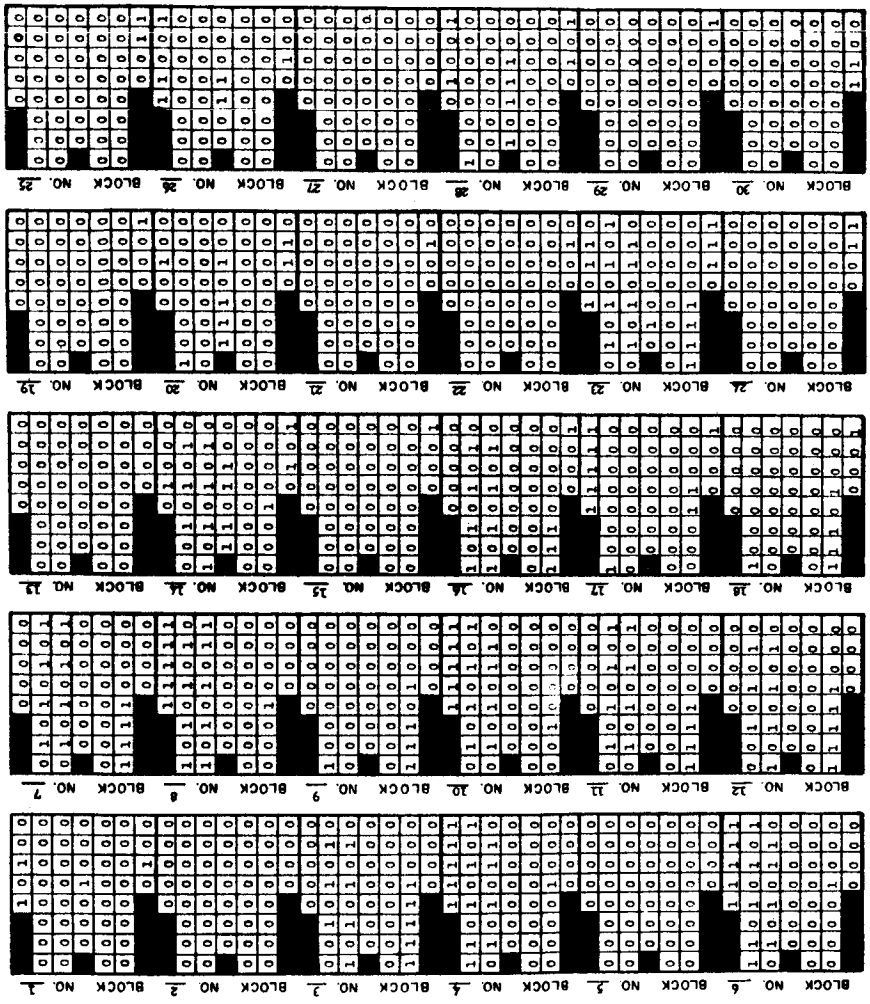


FIGURE 4.8 CONTROL TAPE INFORMATION FOR GENERATION OF THE SPIRAL

In these examples, no attempt was made to achieve a high plotting speed, the control clock frequency being only 150 cps. The plotting speed was limited by the dynamics of the plotting table. A separate test on the path generator alone indicated a reliable operation up to a control clock frequency of 1.6 KC/sec.

The basic interpolation interval for these examples was fixed at eight units. No provision for independent variable scale changing was incorporated in the final implementation due to a lack of sufficient logic in the Digital Synthesizer.

The quantization level of the plotting table has purposely been chosen quite course so that the accuracy of the path generator can be seen. A smaller quanta size would not allow the generator accuracy to be observed as readily.

Discussion of Design Features

The basic interpolation interval h was set for convenience at a nominal value of 8 units. Depending on size and speed requirements, it may be desirable to increase this interval. Following the design procedures outlined in this research, a design for a different basic interval could easily be accomplished.

No attempt has been made here to achieve constant path velocity. A constant component velocity, that of the independent variable, is maintained however. Considering the slope

limitations on generated paths, this means that path velocity is controlled within 41 percent. If more accurate path velocity control is desired, first difference information can be used to determine the average slope over an interval. Using this as an input to a binary rate multiplier, the control clock rate could be regulated to give path velocity control with a maximum variation of 20 percent. For slowly changing paths, the velocity variation would be much less. If better velocity control is desired, more extensive and expensive equipment is needed.[2] Control of path velocity to maintain other parameters such as a cutting tool temperature may be desired.[29] Again, rather extensive equipment is needed.

The integrator units in the path generator of this research have been implemented using flip-flop registers. As the registers become longer to accomodate greater capacity and accuracy, the use of drum storage or delay line storage shows greater economic advantages.[4,10]

The outputs of the path generator have been interpreted in a x-y Cartesian coordinate system. However, this is not a limitation. The outputs could also be interpreted as polar coordinates, i.e. the independent variable output could be interpreted as a polar

angle and the dependent variable output as a radius vector. Continuous contours in which the origin of the coordinate system is inside the contour could then be generated quite easily.[13,14]

The path generator described here has produced inputs for digital servomechanisms, the combination of the generator and the servos serving as a path control system. However, the path generator is by no means limited to such an application. It can also serve to provide information for computing or other control applications.

Summary

The general features of path control systems and the advantages of absolute digital data systems were discussed. A two-dimensional absolute digital data path generator which produces interpolated path control information through discrete input path points was designed and constructed. The generator provided inputs to a digital plotting table, the combination of the generator and the plotter representing a path control system. Details of input data preparation for the system in order to generate two desired paths were presented. The path records produced by the system were given. Examination of these records showed that the desired paths were generated within the designed system error limits.

APPENDIX I

DERIVATION OF THIRD ORDER INTERPOLATION FORMULA

It is desired to find the equation of the central section of the third order polynomial which passes through the four successive points shown in Fig. A. 1. 1. These points are spaced at equal increments, h, of x, the argument. Coefficients of the polynomial are desired in the form of functions of the differences of the values of the polynomial at the points. These differences are defined in Table 2.1.

To start, let y(x) be the polynomial. It may be written in the form

$$y(x) = a_0 + a_1(x - x_0) + a_2(x - x_0)(x - x_1) \qquad \text{(A. 1-1)}$$

$$+ a_3(x - x_0)(x - x_1)(x - x_2)$$

By successively substituting the values of the coordinates of the four points into this equation, the coefficients a_0, a_1, a_2, and a_3 can be evaluated.

$$\text{At } x = x_0, \qquad y(x_0) = y_0 = a_0 \qquad \text{(A. 1-2)}$$

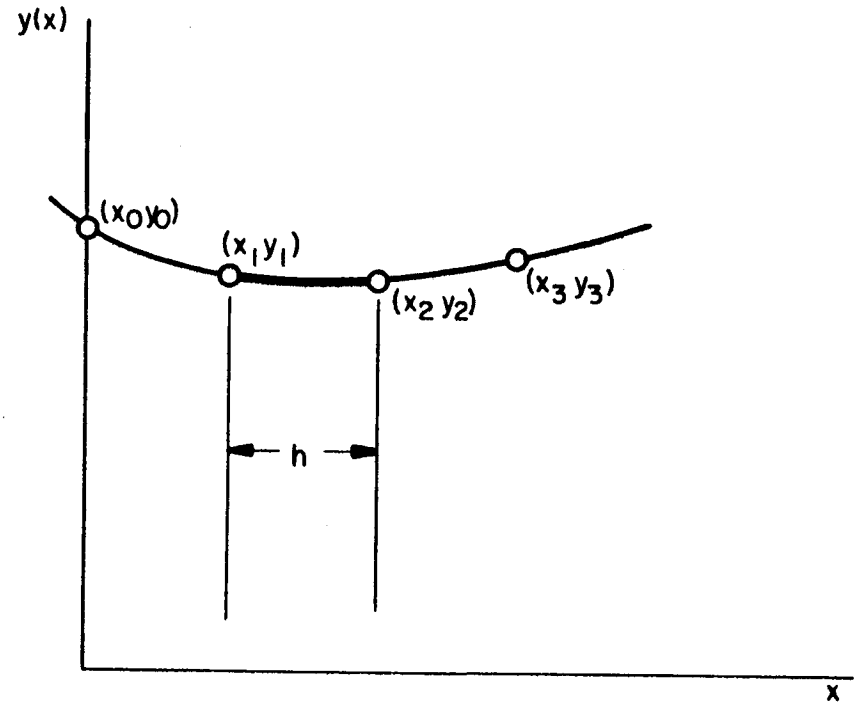


FIGURE A.1.1 POLYNOMIAL DETERMINATION

At $x = x_1$, $\quad y(x_1) = y_1 = y_0 + a_1(x_1 - x_0)$

or, $$a_1 = \frac{y_1 - y_0}{h} = \frac{\Delta y_0}{h} \qquad \text{(A.1-3)}$$

At $x = x_2$, $y(x_2) = y_2 = y_0 + \frac{\Delta y_0}{h}(x_2 - x_0) + a_2(x_2 - x_0)(x_2 - x_1)$

$$= y_0 + \frac{\Delta y_0}{h}(2h) + a_2(2h^2)$$

or, $$a_2 = \frac{y_2 - 2y_1 + y_0}{2h^2} = \frac{\Delta^2 y_0}{2h^2} \qquad \text{(A.1-4)}$$

At $x = x_3$, $y(x_3) = y_3 = y_0 + \frac{y_1 - y_0}{h}(x_3 - x_0)$

$$+ \frac{y_2 - 2y_1 + y_0}{2h^2}(x_3 - x_0)(x_3 - x_1)$$

$$+ a_3(x_3 - x_0)(x_3 - x_1)(x_3 - x_2)$$

$$= y_0 + \frac{y_1 - y_0}{h}(3h) + \frac{y_2 - 2y_1 + y_0}{2h^2}(3h)(2h)$$

$$+ a_3(3h)(2h)(h)$$

or, $$a_3 = \frac{y_3 - 3y_2 + 3y_1 - y_0}{6h^3} = \frac{\Delta^3 y_0}{6h^3} \qquad \text{(A.1-5)}$$

Using the determined values of the a's,

$$y(x) = y_0 + \frac{\Delta y_0}{h}(x - x_0) + \frac{\Delta^2 y_0}{2h^2}(x - x_0)(x - x_1)$$

$$+ \frac{\Delta^3 y_0}{6h^3}(x - x_0)(x - x_1)(x - x_2) \qquad \text{(A.1-6)}$$

This is the equation of the polynomial through the four points.

Now consider only the central section, i.e. let $x = x_1 + \alpha h$

where $0 \leq \alpha \leq 1$.

Then, $$y(x_1 + \alpha h) = y_0 + \frac{\Delta y_0}{h}(x_1 + \alpha h - x_0)$$

$$+ \frac{\Delta^2 y_0}{2h^2}(x_1 + \alpha h - x_0)(x_1 + \alpha h - x_1)$$

$$+ \frac{\Delta^3 y_0}{6h^3}(x_1 + \alpha h - x_0)(x_1 + \alpha h - x_1)(x_1 + \alpha h - x_2)$$

Simplifying,

$$y(x_1 + \alpha h) = y_0 + \Delta y_0 (\alpha + 1) + \frac{\Delta^2 y_0}{2}(\alpha + 1)(\alpha)$$

$$+\frac{\Delta^3 y_0}{6}(a+1)(a)(a-1)$$

$$= y_0 + \Delta y_0 + a\,\Delta y_0 + \frac{\Delta^2 y_0}{2}(a+1)(a)$$

$$+\frac{\Delta^3 y_0}{6}(a+1)(a)(a-1)$$

$$= y_1 + a(y_1 - y_0) + a(y_2 - 2y_1 + y_0)$$

$$+\frac{\Delta^2 y_0}{2}(a)(a-1) + \frac{\Delta^3 y_0}{6}(a+1)(a)(a-1)$$

Finally,

$$y(x_1 + ah) = y_1 + a\,\Delta y_1 + \frac{\Delta^2 y_0}{2}(a)(a-1) \qquad \text{(A.1-7)}$$

$$+\frac{\Delta^3 y_0}{6}(a+1)(a)(a-1)$$

This is the same as the Newton-Gauss central difference interpolation formula when fourth and higher order differences are zero, as they would be for a third order polynomial.[17,24,30]

APPENDIX II

DETAILED DESCRIPTION OF PATH GENERATOR

The purpose of this section is to describe the design and construction details of the digital path generation unit which has been shown in block diagram form in Fig. 3.3. The main areas of discussion will be the path generator, the control logic, and the input tape equipment.

Path Generator

The path generation unit was constructed using the Digital Synthesizer of the Case Digital Systems Laboratory. See the photograph of Fig. A.2.1. The Synthesizer contains transistorized digital logic modules mounted on racks inside the main cabinet which is shown at the left. The input and output connections of the modules are brought out to the receiver pictured at the lower right. Power supply wiring is contained internally. A removable board which fits into the receiver unit contains the desired connecting plug-in wiring. A small receiver board which contains facilities for connecting in auxilliary logic modules and also contains the indicator light connections is shown on the extreme right. Jacks for providing external inputs and outputs to the

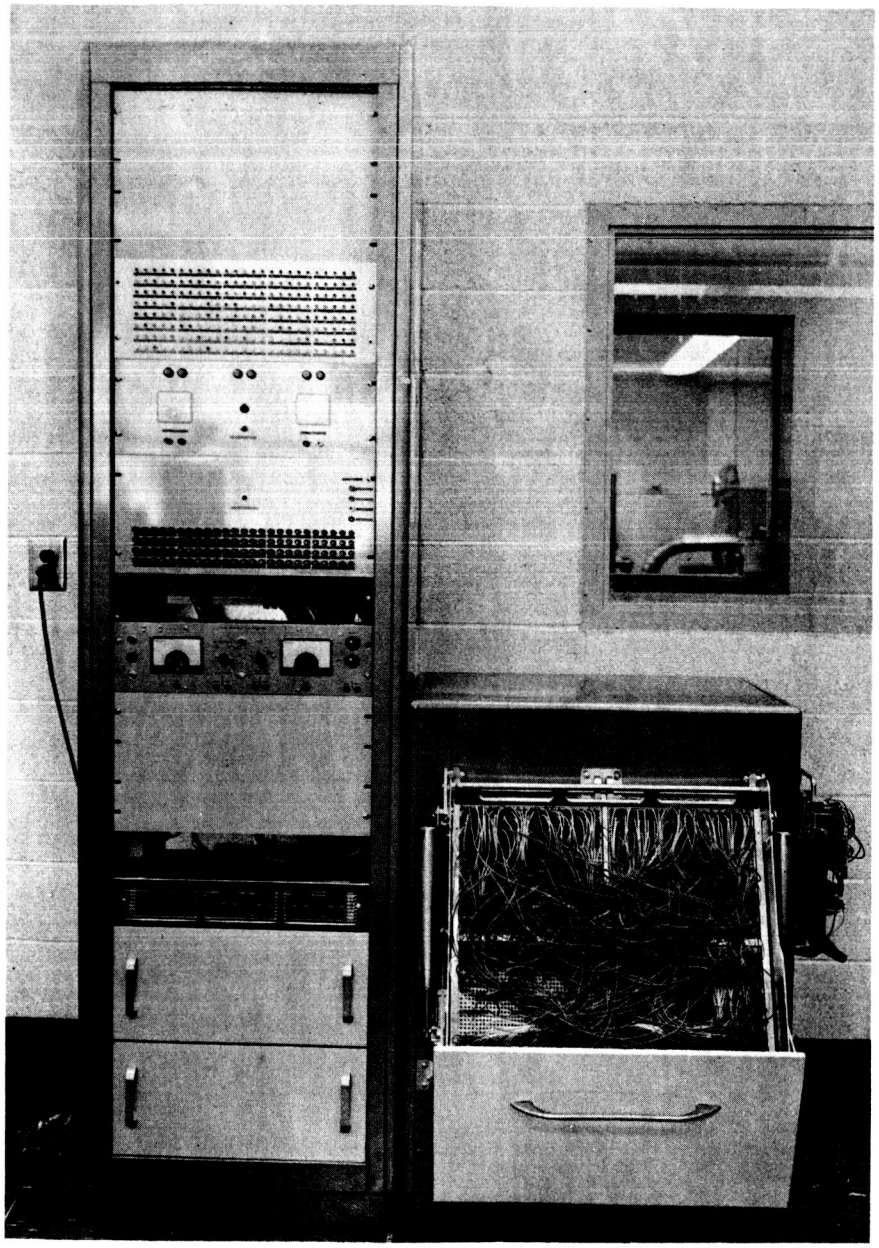


FIGURE A.2.1 PHOTOGRAPH OF DIGITAL SYNTHESIZER

Synthesizer along with indicator lights and a pulse generator are mounted in the main cabinet.

The use of removable wired program boards makes the Synthesizer a valuable research tool. Merely by switching boards the connection of the logic modules needed for one project can be changed to the connection needed for another one.

The logic elements contained in the Synthesizer were manufactured by Wang Laboratories, Inc. They are designated as Series 200 LOGIBLOC Transistorized Module Building Blocks. The maximum operating frequency of these elements is 200 KC. The schematic diagrams for the circuits used in constructing the path generation unit are given in Figs. A.2.6 and A.2.7.

The connection of the logic elements of the Synthesizer used to construct the path generator is shown in Fig. A.2.2. The binary weightings of the flip-flops of the various registers are labeled. The numbered points are connected to the corresponding points of the control logic diagram of Fig. A.2.3.

The registers are connected as arithmetic units of the same type as shown in Fig. 2.7. Here gated pulse generators are used as the delay elements. When the level input is at 0v, the gated pulse generator produces a negative pulse output upon a 1 to 0 transition of the input flip-flop. This pulse is OR gated to the T

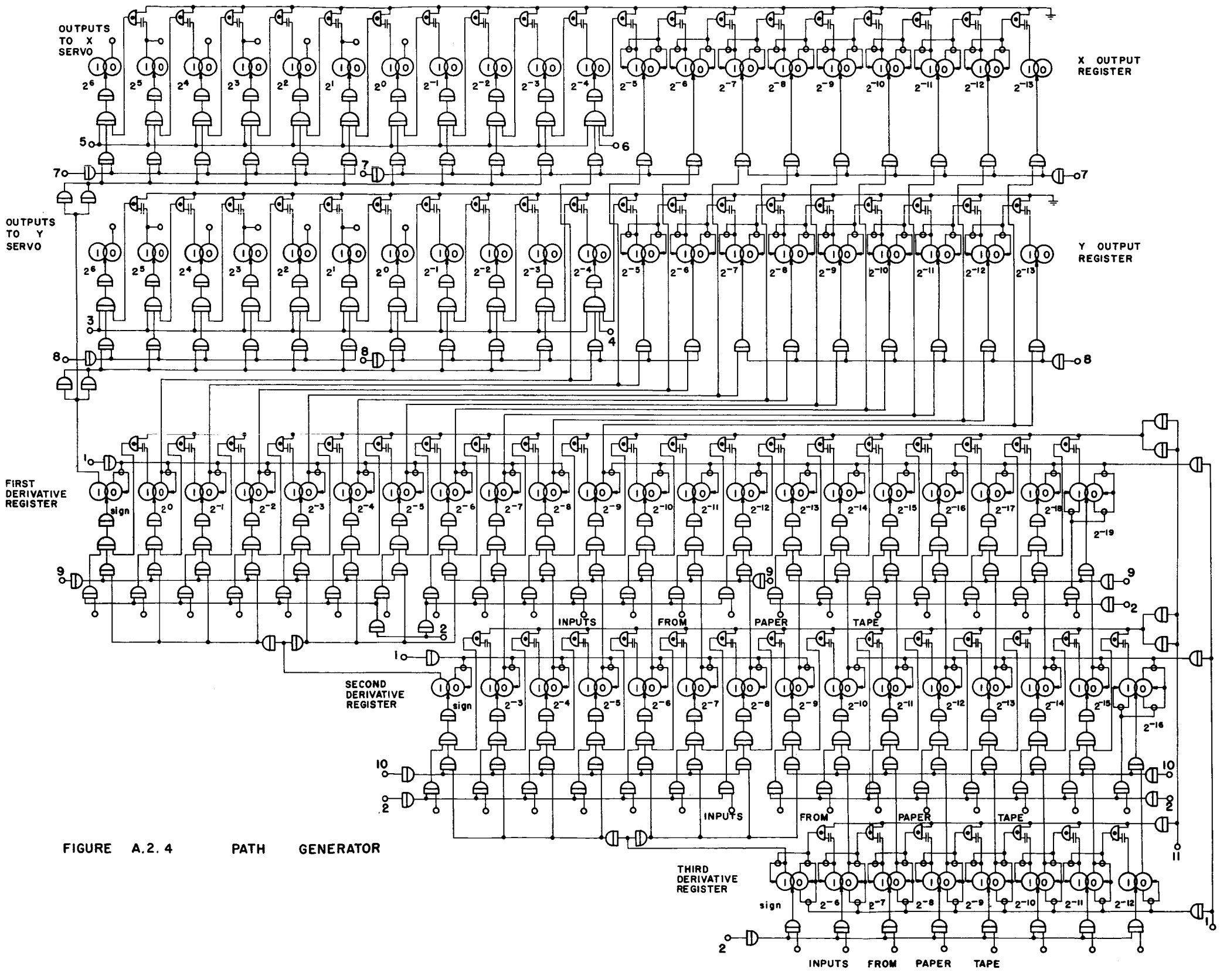


FIGURE A.2.4 PATH GENERATOR

input of the next stage flip-flop. Since a flip-flop changes state upon receiving a positive going pulse or level change, the trailing edge of the pulse from the gated pulse generator provides the trigger. Thus there is a delay of the width of this pulse or about 4.7 μ sec.

If the level input to a gated pulse generator is -12v, all outputs are prevented. Thus the level input can be used for carry inhibiting. This feature is used when it is desired to clear the derivative registers.

Most of the flip-flops receive a T input from a cascaded NOR 3 gate and NOR 1 gate. The combination of the two NOR gates forms an OR gate. Instead of cascading two NOR gates to form an OR gate at the T input of other flip-flops, two additional GATES are connected in as shown. This, in effect, gives a double T input and is the same as an OR gate.

Additional inputs to the OR gates come from NOR 2 gates. For these gates, -12v is normally considered a logical 1 and 0v a logical 0. By reversing this nomenclature, the NOR gate becomes a NAND gate. Use of the trailing edge of the output of the NAND gate allows it to be interpreted as an AND gate.[9] So the NOR 2 gates in effect become AND gates.

There is a row of NOR 2 gates below the x register and also a row below the y register which allows the contents of the first derivative register to be added in. Which of the output registers receives the contents of the first derivative register is determined by the control logic which activates the proper gating row.

A row of NOR 2 gates below the first derivative register is used to add in the second derivative register. A row below the second derivative register allows the third derivative register to be added in. Also, below each derivative register is a row of NOR gates which are used to add in contents from tape. The various add lines are activated by the control logic to allow the control flow diagram of Fig. 3.4 to be followed.

The independent variable scaling feature mentioned in Chapters II and III has not been included in the final implementation of the path generator due to a lack of sufficient logical elements in the Digital Synthesizer. Thus the magnitude of the independent variable increment is fixed at 2^{-4} units. A positive increment is then added at the 2^{-4} stage of the x and y register. A negative increment is added in two's complement form. The address and sign of the independent variable supplied by the input tape allows the control logic to activate the proper gating line.

Variable scaling could be achieved by reading the desired value of the independent variable increment to be used over an interpolation interval into a storage register. The contents of this register could then be added to the independent variable register for each independent variable increment and command.

The derivative registers are cleared by supplying a positive going pulse to the GATES connected to the reset side of each of the flip-flops. Any carrys generated by this action are not propogated through the delay elements since previous to this the level input to each gated-pulse generator is changed from 0 to -12v. After the clearing has taken place, the level input is returned to 0v.

The first seven stages of the x and y registers provide the outputs to the plotting table. The outputs are taken off alternate sides of the flip-flops in order to allow their direct use as inputs to a comparator unit. This is explained in detail in Appendix III.

Notice that additional non-logical elements are included in the implementation of the path generator to compensate for loading effects. Also notice the way in which the sign bit is used to extend the register length so that the proper two's complement representation of negative numbers is presented to the next register.

Control Logic

The control logic of Fig. A.2.3 implements the control flow diagram of Fig. 3.4. This logic consists basicly of two gated counters, one to provide control of the inputs to the generator and one to provide the cyclic addition commands to the integrators units. A cycle control flip-flop governs which counter receives pulses from the pulse generator. At the start of each path, all the counter flip-flops are set to 0, the cycle control flip-flop is set to allow pulses to pass on to the input control counter and the stop control flip-flop is set to allow passage of pulses.

This initializing is done by open circuiting the emitter of the transistor of the appropriate side of a flip-flop using the reset button of the Synthesizer. The initial conditions to the x and y registers are also set in at this time by the same means.

By using trailing edge triggering and the nomenclature that 0 volts represents a logical 1 and -12 volts represents a logical 0, the NOR gates connected to the counters function as AND gates.[9] An increasing level change output is then produced by the gate as a counter leaves a particular count. Thus, gates a, b, c and d produce outputs as the count in the input control counter leaves 1, 3, 5 and 7 respectively. An output is produced by gates e, f and

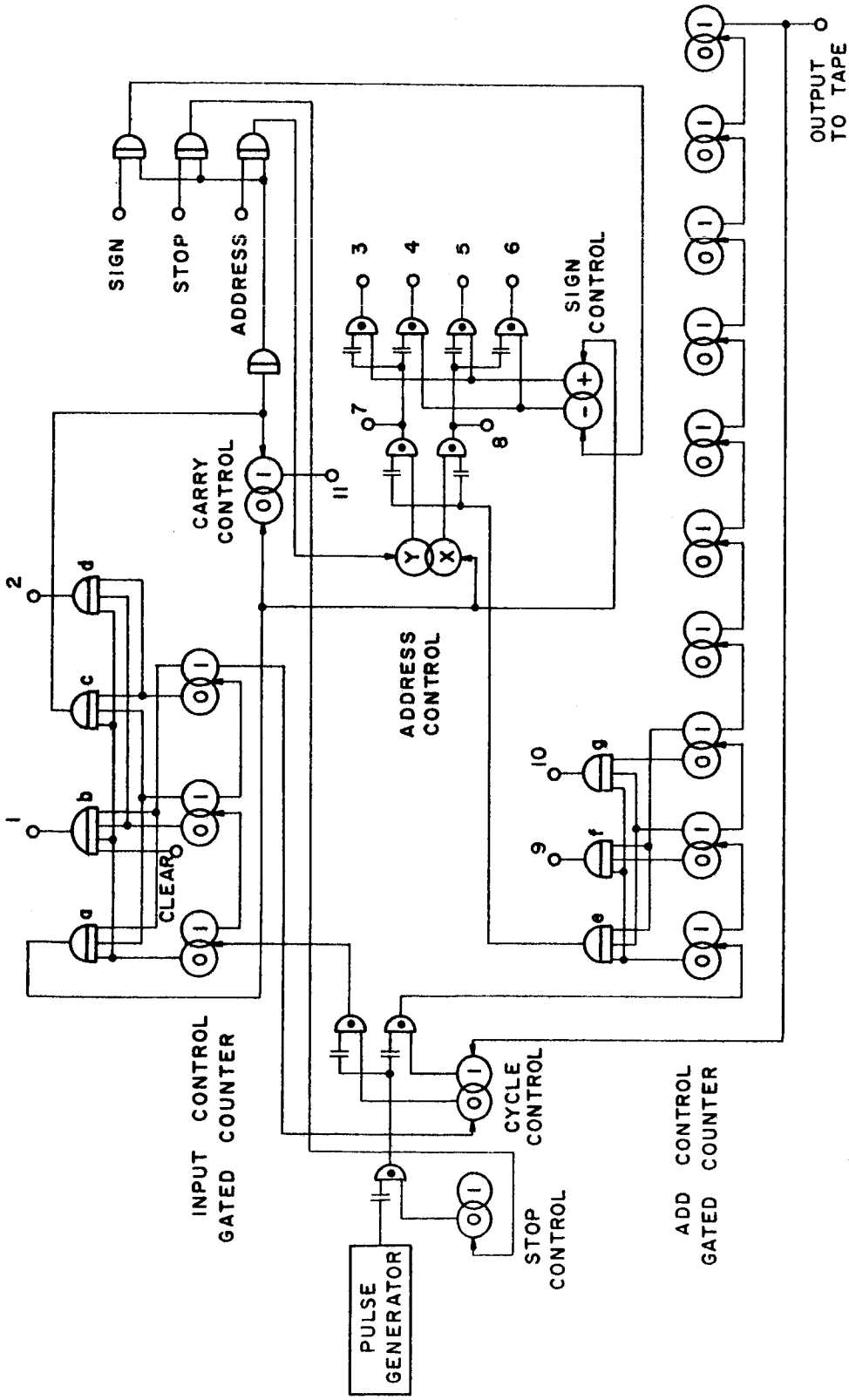


FIGURE A.2.3 CONTROL LOGIC

g as the add control counter leaves the counts 1 modulo 8, 3 modulo 8, and 5 modulo 8 respectively.

The output of gate a resets the carry control flip-flop to prevent any propogation of carrys between stages of the derivative registers. It also sets the independent variable address control flip-flop to x and the independent variable sign control flip-flop to plus. Gate b provides a clear signal to the derivative registers if the clear hole on the input tape is punched. Otherwise no clear signal is produced.

The output of gate c sets the carry control flip-flop to again allow propogation of carrys in the derivative registers. Also the sign, address, and stop control inputs from the input tape are gated in. Gate d provides the signal to add inputs from the tape to the derivative registers.

The 1 to 0 transition of the last flip-flop of the input control counter resets the cycle control flip-flop to route pulses to the add control counter.

The output of gate e is used to add the increment of the independent variable to the independent variable register and to add the first derivative register to the dependent variable register. The address control flip-flop determines which is the independent variable register. The sign control flip-flop gives

the sign of the independent variable increment. These two flip-flops then control the necessary gating.

Gate f provides the signal to add the contents of the second derivative register to the first derivative register. The output of gate g causes the addition of the third derivative register to the second derivative register.

For the add control counter, those stages beyond the first three serve as in iteration counter for the add cycle. Since 2^4 add cycles must take place to cause the independent variable to increase one unit, and 2^3 units are needed for an interpolation interval, 2^7 add cycles must take place for each interpolation interval. Thus, seven stages beyond the three needed for the gating of one add cycle are needed to count the iterations. The add control counter then has ten total stages.

The 1 to 0 transition of the last stage of the add control counter or overflow of the counter means that sufficient add iterations have occurred for an interval. Thus it is used to set the cycle control flip-flop to again route pulses to the input control counter. The 0 to 1 transition of the last stage flip-flop provides the level change input to cause the tape block reader to advance one block length. This transition occurs half way

through the capacity of the add control counter and allows the tape transient to be completed before a new reading cycle takes place.

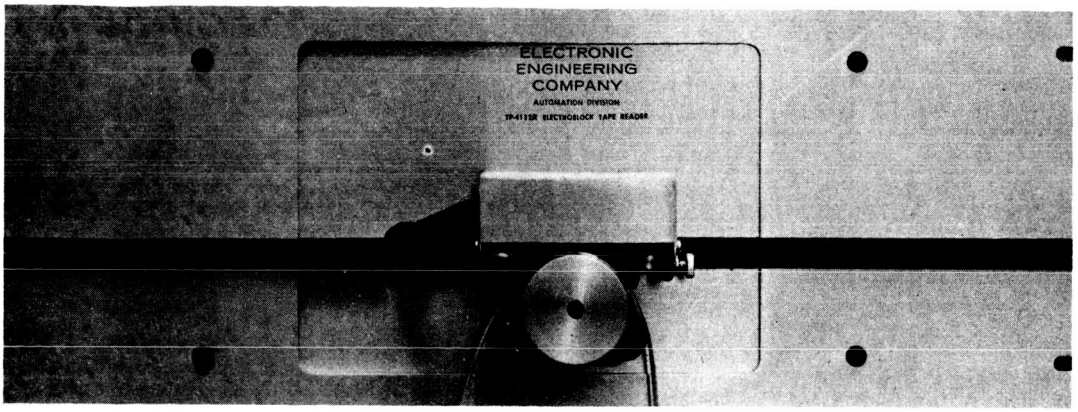
The interchange between the operation of the input control counter and the add control counter continues until a stop signal is punched on the tape. During the input cycle in which this signal appears, the stop control flip-flop is set to prevent further pulses from reaching the control counters and thus stops the operation of the path generator.

Tape Block Reader and Tape Punch

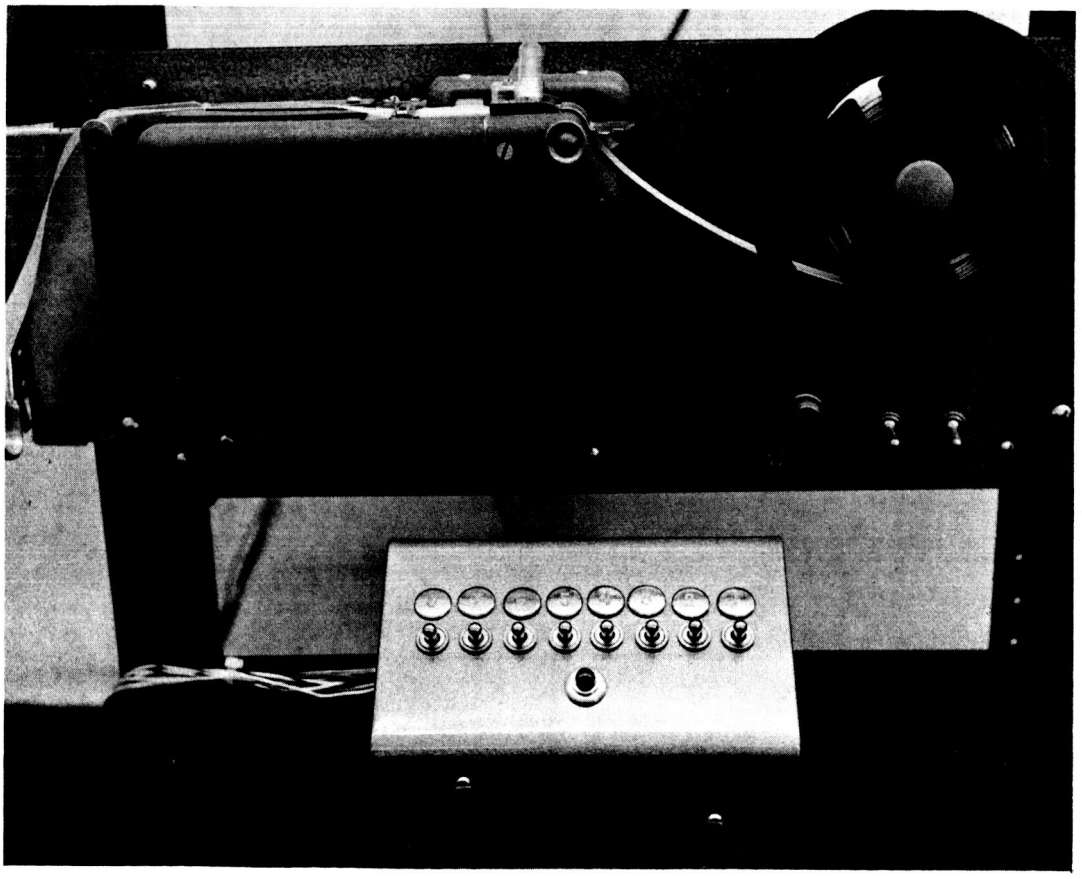
The input tape is read by Model 4112-R ELECTROBLOK Reader manufactured by the Electronic Engineering Co. of California. See the photograph of Fig. A.2.4a. This reader features a stepping motor drive which advances the tape one block length for each +25v trigger pulse input. The 0 to -12 volt level change produced by the control logic is used as the input to the circuit shown in Fig. A.2.5b. This circuit then produces the appropriate magnitude trigger pulse.

The block reader actually is capable of reading 12 lines of 8 level tape. However, only seven of these lines are needed and used to provide the input information to the path generator.

Each output brush is connected by the circuit shown in Fig. A.2.5a to the appropriate input gate of the path generator. Thus,



A) TAPE BLOCK READER



B) TAPE PUNCH

FIGURE A.2.4 PHOTOGRAPHS OF TAPE EQUIPMENT

a punched hole on 1 on the tape causes a brush to contact the common cylinder producing a ground level output. A blank or 0 on the tape insulates the brush from the common cylinder and produces an open circuit voltage of -12 volts. The loading of this circuit by an input gate, of course, reduces the magnitude of this voltage.

The block reader has the feature that the line connecting the common cylinder to a voltage level, in this case ground, is opened during tape transient. This prevents the arcing of the brushes.

The input tape to the reader is prepared using the tape punch shown in the photograph of Fig. A.2.4b. The punching unit is controlled by a console containing a switch for each of the eight levels of tape plus a single line punch button. The tape feed hole is automatically punched each time the punch button is activated. Other holes are punched by turning on the appropriate column switch prior to pushing the punch button.

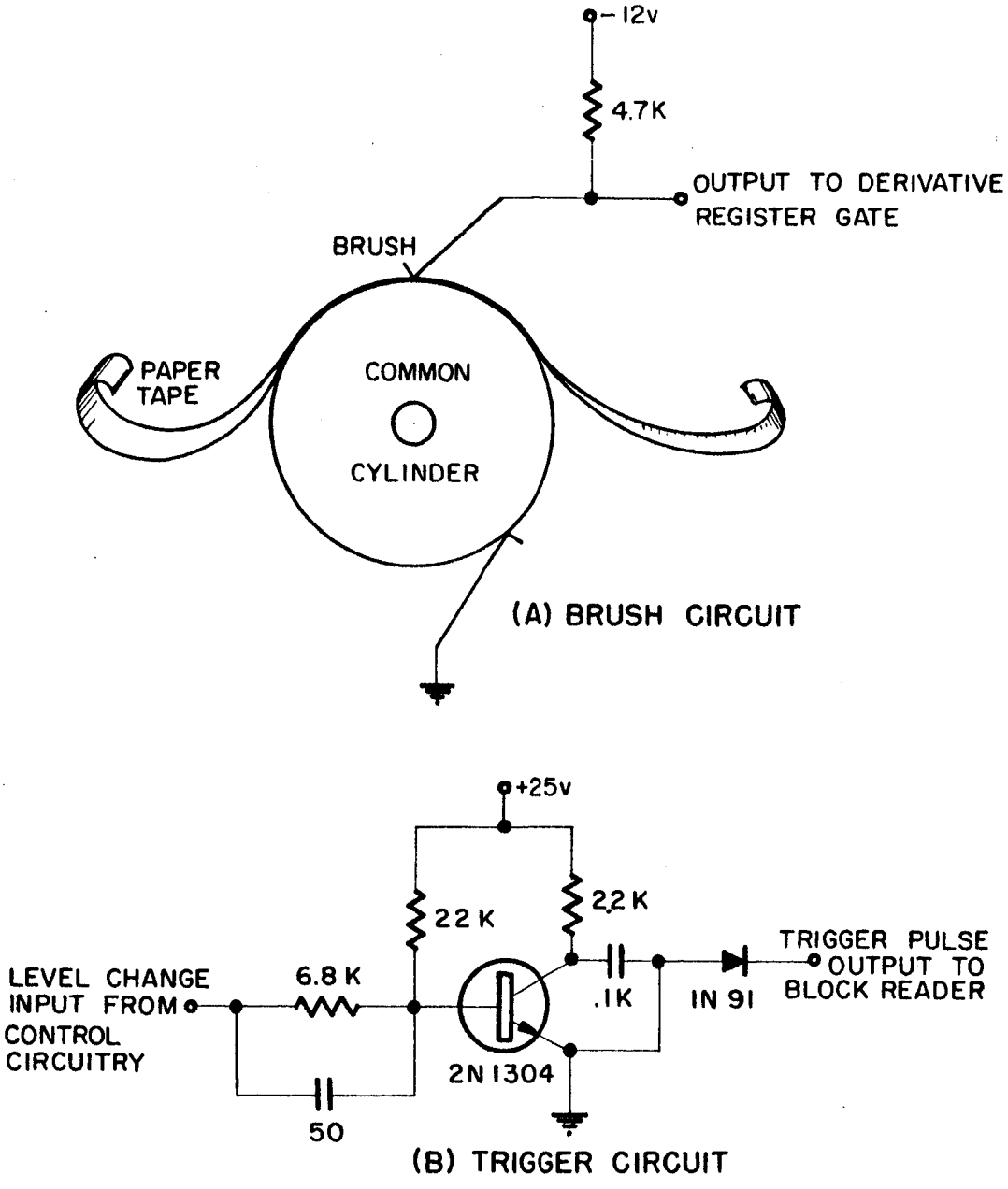


FIGURE A.2.5 – TAPE READER CIRCUITS

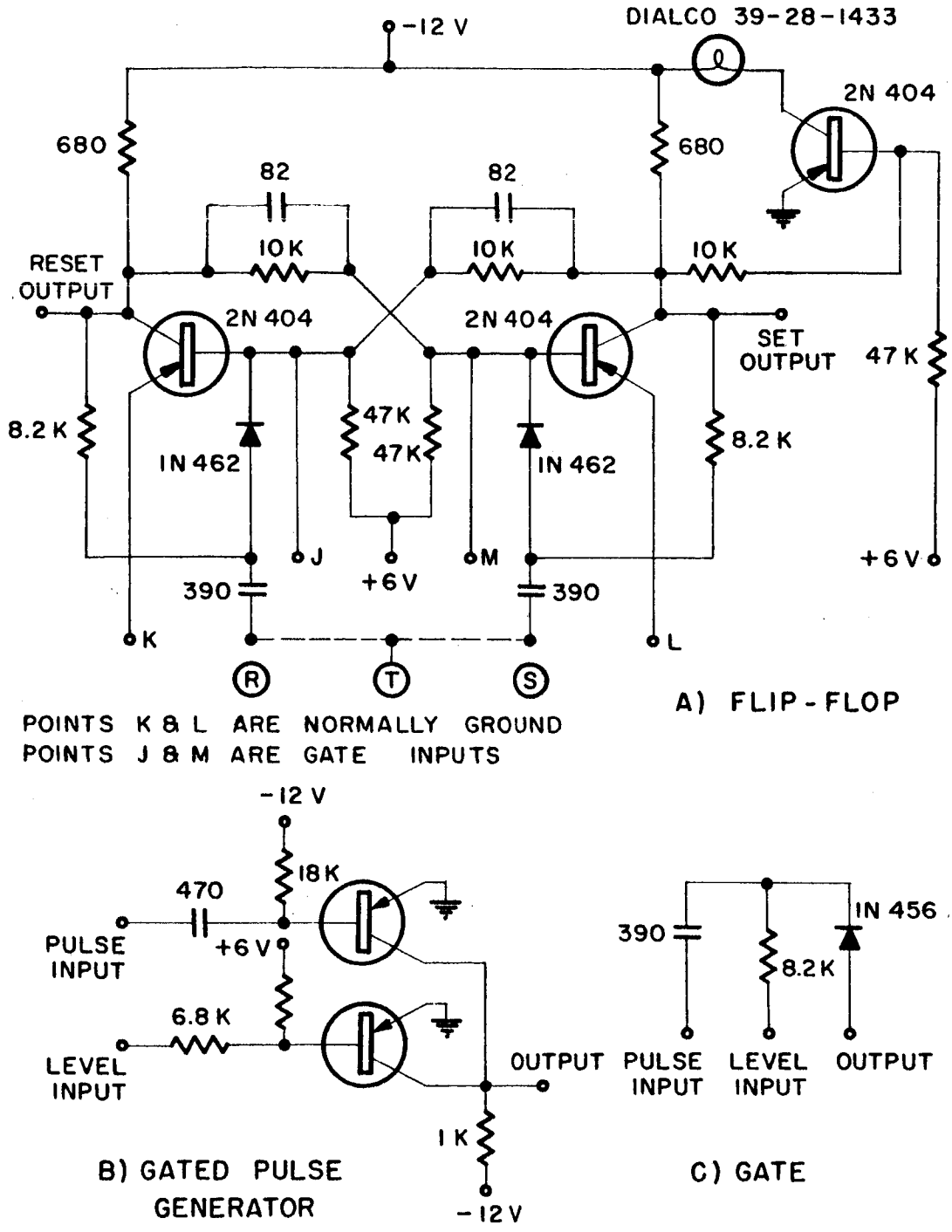
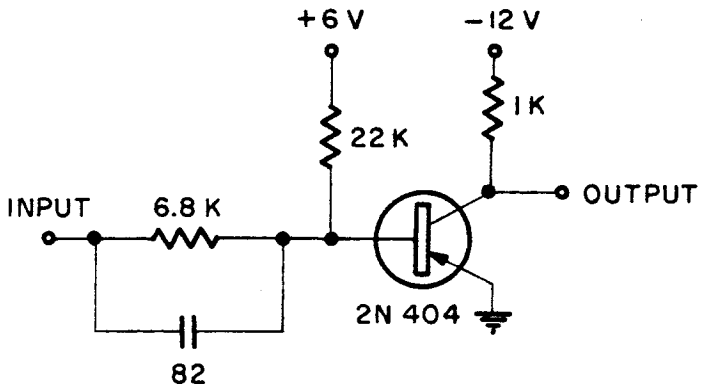
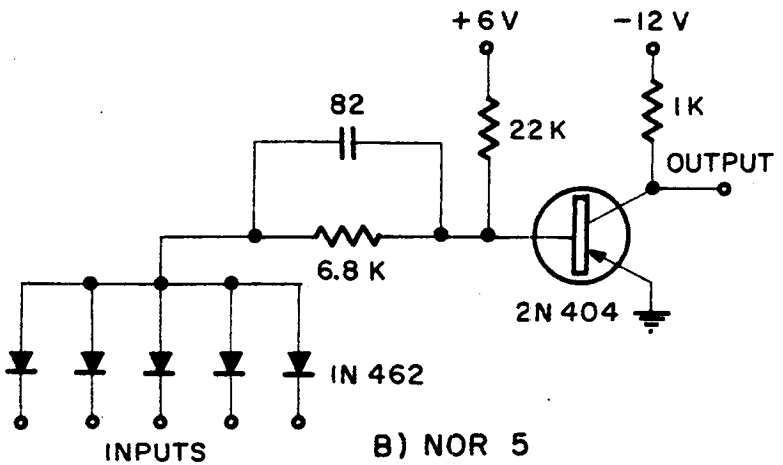


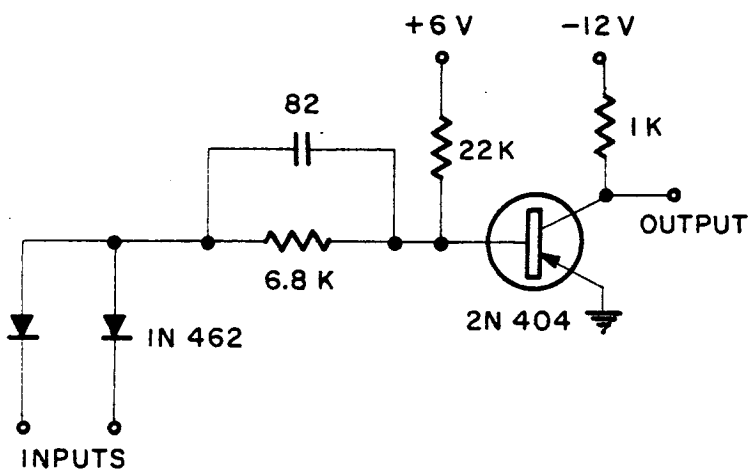
FIGURE A.2.6 – LOGIC MODULE CIRCUITS



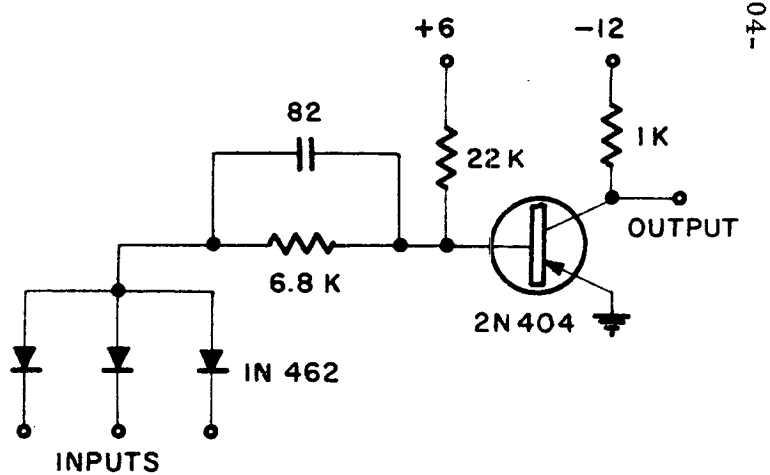
A) NOR 1



B) NOR 5



C) NOR 2



D) NOR 3

FIGURE A.2.7 – LOGIC MODULE CIRCUITS

APPENDIX III

DETAILED DESCRIPTION OF THE CONTROLLED SYSTEM

The purpose of this section is to describe in detail the two-dimensional control system which accepts the absolute digital path control information from the input generator and converts this information into a graphical record of the generated path. The constructed system is composed of the following main elements:

1. Two-dimensional crossbar plotter,
2. Feedback encoders and read-out logic,
3. Comparators, decoders and modulators, and
4. Servo amplifiers, motors and gear trains

This system is represented in block diagram form in Fig. A.3.1. General operation of such a system was discussed in Chapter I. The actual controlled system is shown in the photograph of Fig. A.3.2. The operation of each element will now be described in detail.

Two-dimensional Crossbar Plotter

In order to provide a continuous graphical output record, a two-dimensional plotter was constructed. See Fig. A.3.3. This plotter represents the system which is to be controlled by the

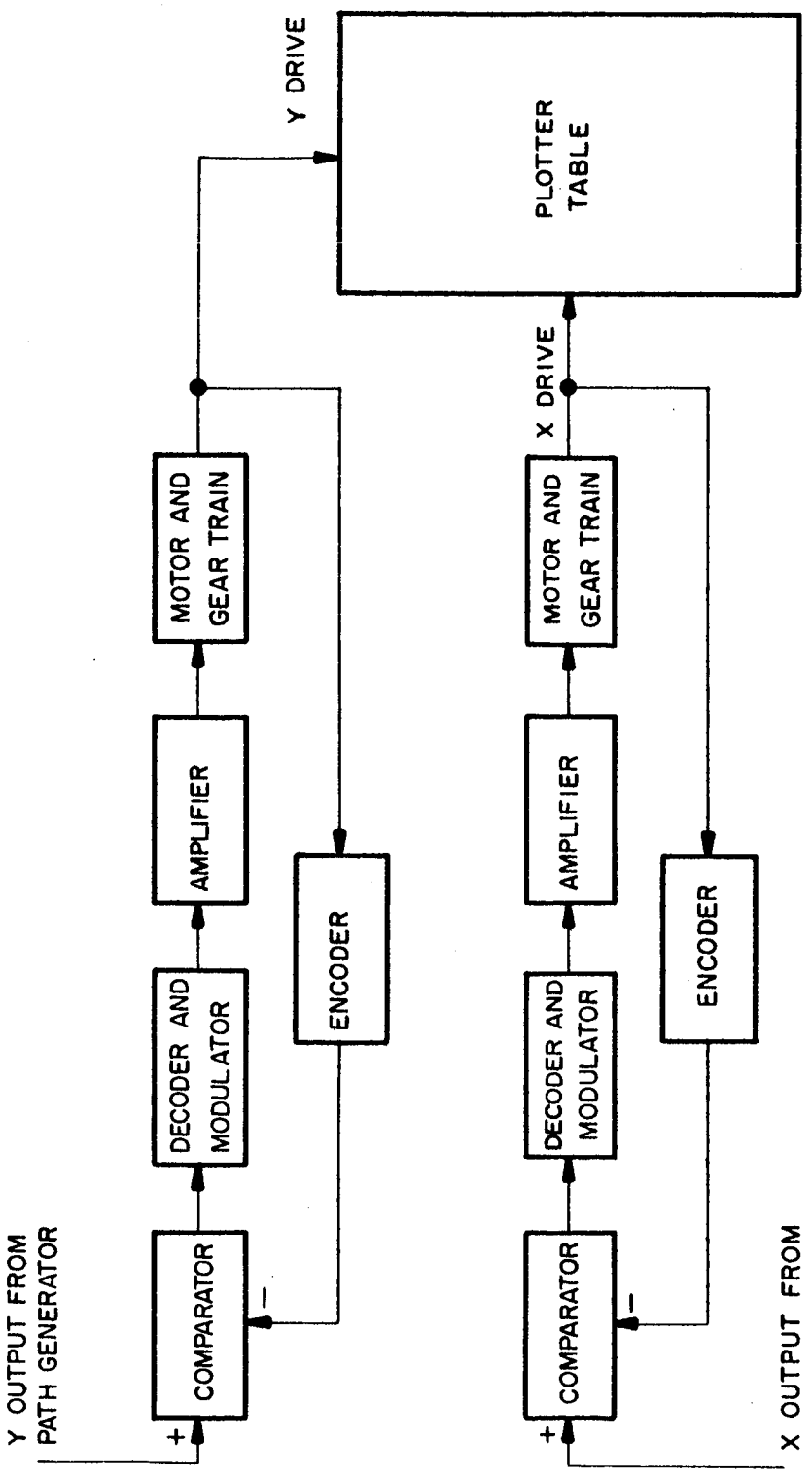


FIGURE A.3.1 – BLOCK DIAGRAM OF CONTROLLED SYSTEM

FIGURE A.3.2 PHOTOGRAPH OF CONTROLLED SYSTEM

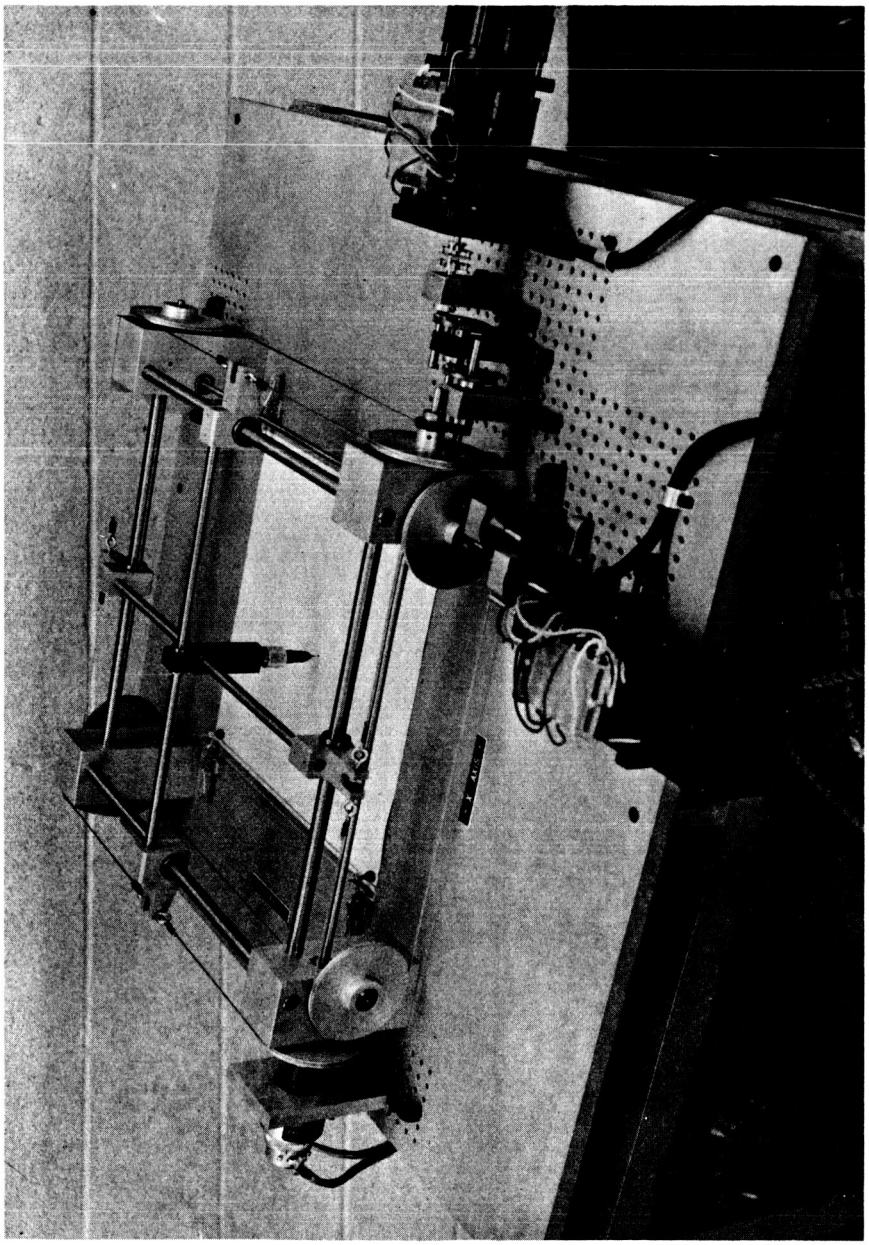


FIGURE A.3.3 PHOTOGRAPH OF TWO-DIMENSIONAL PLOTTER

input generator. In practice this plotter would be replaced by the actual physical system which is to be controlled. This physical system could be a machine tool, an automatic drafting machine, an orthotic arm aid, or any other device requiring two-dimensional path control.

The plotter is based upon a crossbar mechanism. A follower device which contains a recording pen is driven by two shafts which are at right angles to one another. These shafts pass through linear ball bushings which are mounted in the follower device. These two right angle shafts provide the x and y movements of the follower. Thus by simultaneously controlling the positions of these shafts the follower can be made to generate any two-dimensional contour.

The follower control shafts are moved by cables attached to brackets at the ends of the shafts. These cable brackets contain linear ball bushings which ride on shafts mounted in the corner blocks of the system. Each cable passes over a drive pulley and an idler pulley.

The two drive pulleys for each coordinate direction are connected by a drive shaft. Each drive shaft is geared to a motor. A bellows coupling connects each drive shaft directly to an encoder.

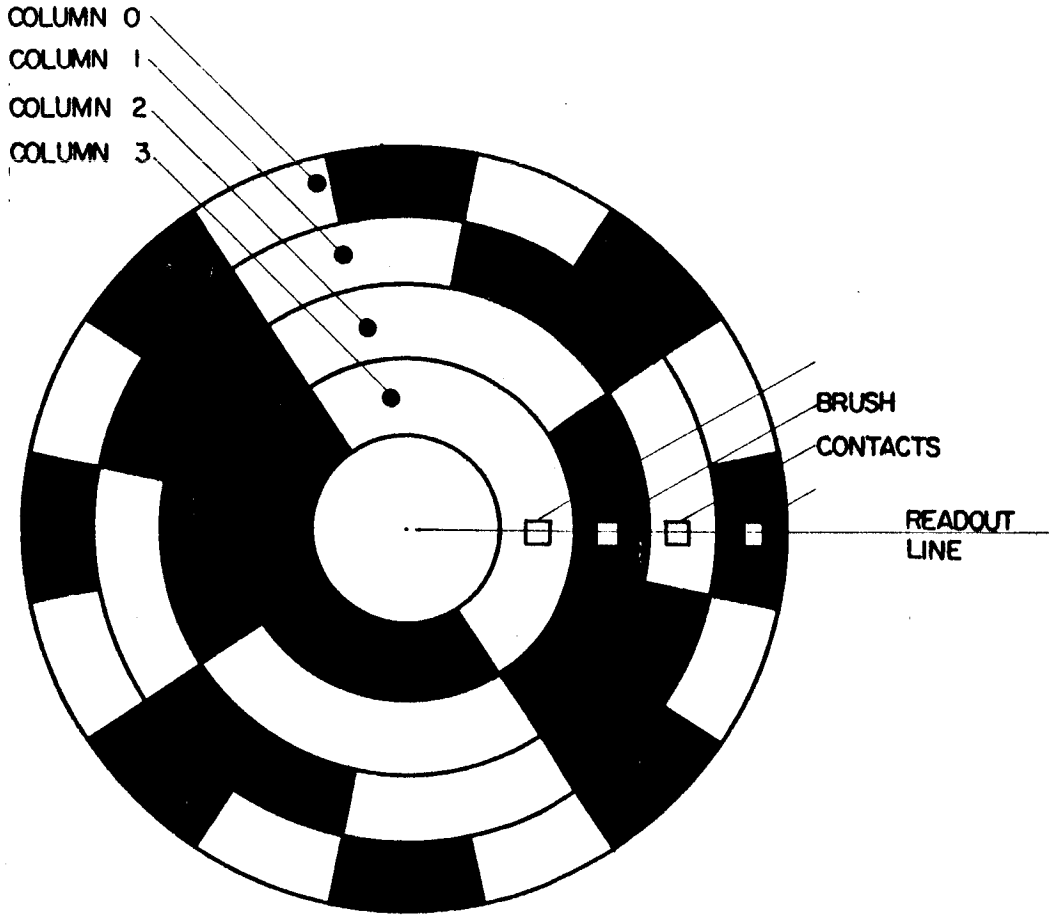
The encoders used in this system produce a seven bit output. Thus any of 2^7 or 128 distinct outputs are available. Each quanta zone was chosen to represent 0.080 inch linear travel along an axis. Thus the length on any axis represented by one rotation of the encoder is 128 x 0.080 = 10.240 inches. The plotter thus can construct a contour over a 10.240 inch square area within an accuracy of 0.080 inch on any axis. In order that the encoders could be driven directly by the drive shafts, thus eliminating any gearing and resultant backlash possibilities, and also in order that one encoder revolution would represent the full linear travel of an axis, the pitch circumference of the drive pulleys was chosen as 10.240 inches giving a pitch radius of 1.630 inches.

The plotter was constructed so that the x and y coordinate inertias and drives are symmetric. Thus similar dynamic properties can be expected in the two dimensions.

Feedback Encoders and Readout Logic

The analog feedback positions are converted to digital representations by Librascope No. 707 V-brush binary encoders.

The basic part of this encoder is a disk with a binary code pattern as shown in Fig. A.3.4. If brushes are placed on line as shown, the binary number representing the position of the disk can



NOTE: SHADED AREAS REPRESENT
BINARY 1'S

FIGURE A.3.4 – BINARY CODED DISK

be read from the brushes. In this example the coded pattern array is arranged in a series of four zones, corresponding to the capability of sensing four binary digits. The present position of the disk represents the binary number 0101.

However, the arrangement of brushes and zones as shown in Fig. A.3.4 will not work in practice because of the ambiguity which can occur in reading the coded pattern. To see this consider the rotation of this sample disk from decimal 7 (0111) to the decimal 8 (1000) position. All the brush contacts must change their contacting status simultaneously.

This would require infinitely narrow brushes as well as perfect alignment of both the brushes and the code zones. The physical limitations obviously prevent any ideal transition and some brushes change before others. For the natural binary code, this can lead to some serious reading errors.

The V-brush method provides for reading out unambiguously a binary coded disk. The logic for such a system can be deduced from observation of the nature of the binary code. The sequence of four bit binary numbers is given in Table A.3-1.

Examination shows that when the least significant digit changes from 0 to 1 in the direction of increasing count none of the other digits change. Further, when the least significant digit

TABLE A.3-1

SEQUENCE OF FOUR BIT BINARY NUMBERS

Binary Number					Decimal Equivalent	
Position Number, m	3	2	1	0	1	0
	0	0	0	0		0
	0	0	0	1		1
	0	0	1	0		2
	0	0	1	1		3
	0	1	0	0		4
	0	1	0	1		5
	0	1	1	0		6
	0	1	1	1		7
	1	0	0	0		8
	1	0	0	1		9
	1	0	1	0	1	0
	1	0	1	1	1	1
	1	1	0	0	1	2
	1	1	0	1	1	3
	1	1	1	0	1	4
	1	1	1	1	1	5

changes from 1 to 0 in the increasing count direction, the digit in the next most significant column always changes. Generalizing, an increasing transition from 0 to 1 in the mth row always causes a change in the mth +1 digit.

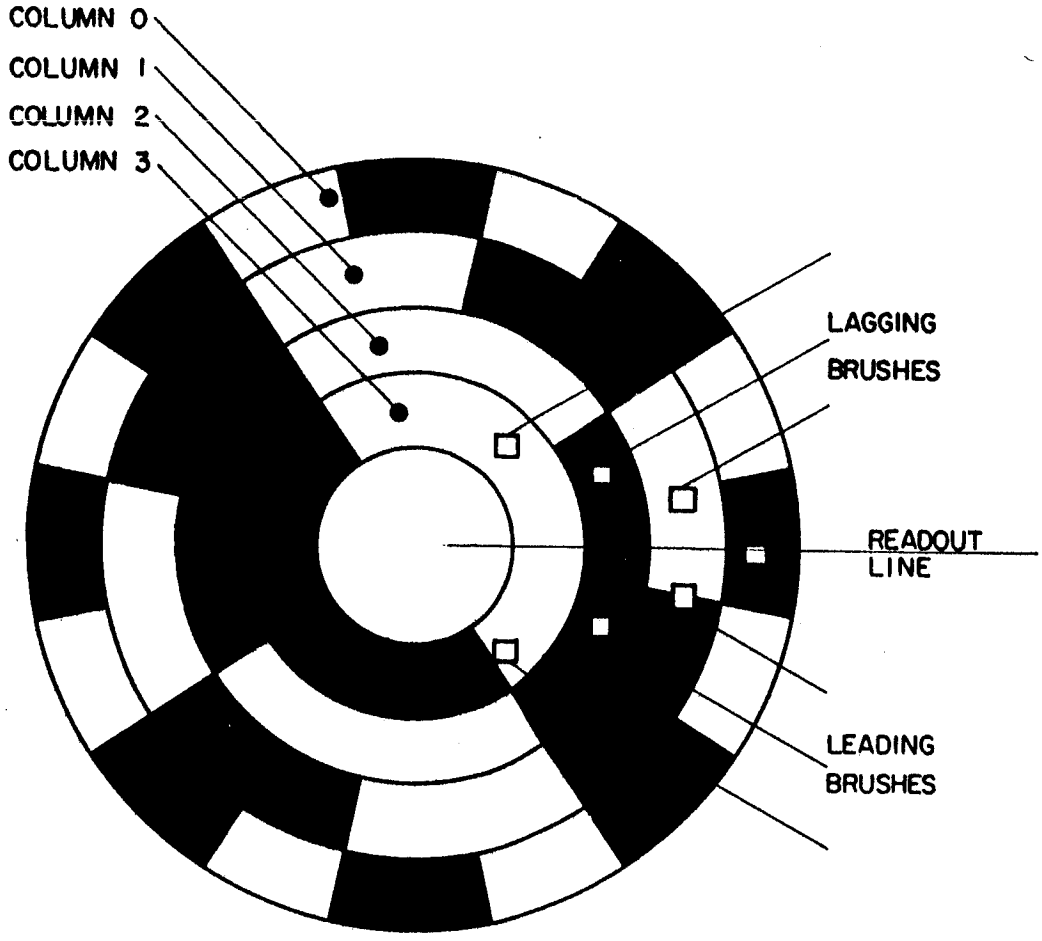
A further observation can be made from Fig. A.3.4. Considering an increasing count, when the least significant digit is 1, the count in the next column has not changed for a minimum of the width of the least significant digit. When the least significant digit is 0, the count in the next column will not change for a minimum of the width of the least significant digit. Generalizing, when a digit in the mth column is 1, the digit in the m + 1 column has not changed for a minimum of 2^m rows and similarly when a digit in the mth column is 0 the digit in the m + 1 column will not change for a minimum of 2^m rows.

These conclusions lead to a method of reading a natural binary code pattern unambiguously. If a 1 is read in the least significant digit, then in the increasing count direction no change in the next digit has occurred recently. If a 0 is read in the least significant digit, a change has occurred recently in the next digit. Therefore, if two brushes were to read the next digit, one leading and one lagging the reading line, any recent change or absence of change could be read. Leading refers to displacement

in the direction of increasing count; lagging, to displacement in the direction of decreasing count. The leading brush is read if the least significant digit is 0 and the lagging brush if the least significant digit is 1. So the next digit can be accurately read based upon the reading of the least significant digit.

If the least significant column were removed, the code pattern would remain the same with the sector representing a row being twice the size of the original sector. The reasoning that was applied to the least significant digit can now be applied to the next digit and consecutively to the remaining columns of the disk.

The placement of the brushes should be symmetrical about the reading line. The brush spacing for a given digit should be equal to the segment width of the next lower order digit. The brushes then fan out in a V-shaped exponential curve. Fig. A.3.5 shows the placement of the brushes on a four digit disk. For the reading line shown, the number to be read is 0101. Since the least significant brush reads 1, the lagging brush is read in column 1. This brush reads 0 so the lead brush is read in column 2. This brush reads 1 so the lagging brush is selected in column 3, which reads 0. Note that each brush that is read is well within the segment which is to be read, eliminating ambiguity.



NOTE: SHADED AREAS REPRESENT BINARY 1'S

FIGURE A.3.5 — V-BRUSH READOUT

The difficulty with this type of readout is that a separate logic circuit is needed with each column to select the proper brush. The Librascope encoder used has seven columns producing $2^7 = 128$ distinct readout numbers. A brush is grounded for a 1, and is open circuited for a 0. Three stages of the seven stage logic that was used in reading the disk are shown in Fig. A.3.6.

In this circuit the least significant brush is B_0, the leading brush in column 1 is B_1 and the lagging brush $\bar{B}_1$. Similar notation applies to the remaining stages.

The means of brush selection is based upon the inversion property of a grounded emitter transistor amplifier. If B_0 is open circuited, i.e., a 0 is read in column 0, an inverted signal which tends toward -12 volts is fed to the base of the B_1 transistor. A ground signal is fed to the base of the $\bar{B}_1$ transistor. Thus only the signal to the emitter of the B_1 transistor affects the output which appears at the junction of the collectors of the B_1 and $\bar{B}_1$ transistors since only the B_1 transistor can be biased into operation. If B_1 is open circuited corresponding to a 0 being read, the output tends toward -12 volts. If B_1 is grounded the output is ground level.

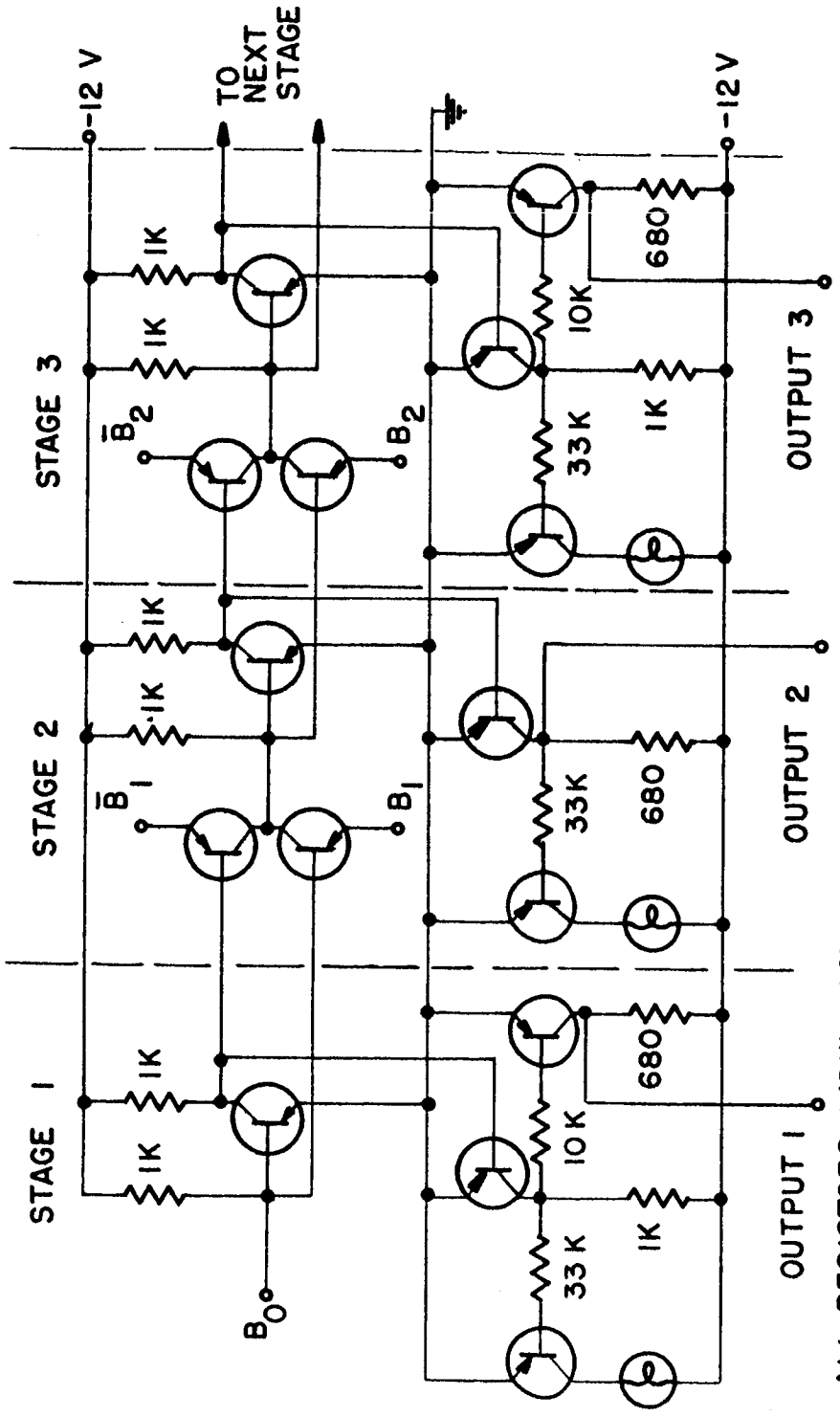


FIGURE A.3.6 – ENCODER READOUT LOGIC

If B_0 is grounded representing a binary 1, a signal tending toward -12 volts appears at the base of the $\bar{B}_1$ transistor and a ground signal appears at the base of the B_1 transistor. Thus only the $\bar{B}_1$ signal can appear at the output collector junctions. Hence, if the input to the first stage is 0, the leading brush of the next stage is read. If the input to the first stage is 1, the lagging brush of the next stage is read. This means of brush selection is propagated through the stages.

In the above explanation it has been stated that a voltage tends toward -12 volts. In each case this signal terminates somewhere in the base of a grounded emitter amplifier. The maximum drop across the base-emitter junction is a few tenths of a volt so that signals which tend toward -12 volts are actually approximately -0.2 volts.

The inverted brush input for each stage is inverted once again by a grounded-base amplifier. The output of this amplifier drives an indicator light and also serves as the output for every other stage. Due to the nature of the comparison operation, every other output must be inverted. At the output of the odd stages, -12 volts represents a 1 and 0 volts represents a 0. The inverse is true for the even stages.

The actual brush selection takes place in serial form. However, the propagation is accomplished with such rapidity that the brush selection may be considered to take place in parallel form. The readout of the stages in parallel form is thus justified.

Comparators, Decoders and Modulators

The comparator is basically an arithmetic subtraction unit which provides the sign and magnitude of the difference between two natural binary coded numbers, the input command number and the encoder feedback number. This unit, however, solves a subtraction problem which is simpler than that encountered in a general purpose computer since in this case both the minuend and the subtrahend are always positive numbers. For a general purpose computer both arguments may be either positive or negative necessitating additional sign and operation determination logic and possible special coding schemes for negative numbers. The digital subtraction operation is reduced to an addition operation by the following means. The one's complement of the minuend is added to the subtrahend. The result is then treated according to the following rules:

1. If there is no overflow carry from the most significant stage, the result is positive and equal in magnitude to the one's complement of the sum.
2. If there is an overflow carry, the result is negative and equal in magnitude to the sum plus 1.

To see how these rules are derived consider subtracting the two inherently positive natural binary numbers A(minuend) and B(subtrahend), i.e., it is desired to find (A) - (B). If every bit of the binary number A is inverted, the one's complement of A is obtained which is given by:

$$A^1 = 2^n - 1 - A$$

where A^1 is the one's complement and n is the number of bits in A. The subtraction operation desired is then:

$$(A) - (B) = 2^n - 1 - A^1 - B$$
$$= 2^n - (A^1 + B) - 1$$

Rearranging,

$$A^1 + B = 2^n - (A - B) - 1 \qquad \text{(A.3-1)}$$

or,

$$A^1 + B = 2^n + (B - A) - 1 \qquad \text{(A.3-2)}$$

There are two possibilities to consider, $A \geq B$ and $A < B$.

1. If $A \geq B$, the difference is positive or zero. Applying Eq. A. 3-1 shows that $A^1 + B$ is then the one's complement of the desired difference. No overflow carry results.
2. If $A < B$, the difference is negative. Eq. A. 3-2 shows that there is an overflow (2^n) and that the correct difference may be obtained by adding 1 to the result of $A^1 + B$.

Thus the rules previously stated for reducing the subtraction operation to one of addition have been derived.

The equations governing the addition operation for one stage for binary numbers can be derived from Table A. 3-2. Here A_i and B_i are the i^{th} addend and augend bits to be added, C_i is the incoming carry from the previous stage, S_i is the i^{th} sum bit, and C_{i+1} is the outgoing carry. The reduced logical equations for the sum and carry bits are:

$$S_i = A_i \oplus B_i \oplus C_i \qquad \text{(A. 3-3)}$$

and

$$C_i = A_iB_i + B_iC_i + A_iC_i \qquad \text{(A. 3-4)}$$

TABLE A.3-2

BINARY ADDITION TRUTH TABLE

A_i	0	1	0	1	0	1	0	1
B_i	0	0	1	1	0	0	1	1
C_i	0	0	0	0	1	1	1	1
S_i	0	1	1	0	1	0	0	1
C_{i+1}	0	0	0	1	0	1	1	1

$$S_i = A_i\bar{B}_i\bar{C}_i + \bar{A}_iB_i\bar{C}_i + \bar{A}_i\bar{B}_iC_i + A_iB_iC_i$$

$$= A_i \oplus B_i \oplus C_i$$

$$C_{i+1} = A_iB_i\bar{C}_i + A_i\bar{B}_iC_i + \bar{A}_iB_iC_i + A_iB_iC_i$$

$$= A_iB_i + B_iC_i + A_iC_i$$

$$\bar{S}_i = \bar{A}_i\bar{B}_i\bar{C}_i + A_iB_i\bar{C}_i + A_i\bar{B}_iC_i + \bar{A}_iB_iC_i$$

$$= \bar{A}_i + \bar{B}_i + \bar{C}_i$$

$$\bar{C}_{i+1} = \bar{A}_i\bar{B}_i\bar{C}_i + A_i\bar{B}_i\bar{C}_i + \bar{A}_iB_i\bar{C}_i + \bar{A}_i\bar{B}_iC_i$$

$$= \bar{A}_i\bar{B}_i + \bar{B}_i\bar{C}_i + \bar{A}_i\bar{C}_i$$

Also note that

$$\bar{S}_i = \bar{A}_i \oplus \bar{B}_i \oplus \bar{C}_i \qquad \text{(A.3-5)}$$

$$\bar{C}_{i+1} = \bar{A}_i\bar{B}_i + \bar{B}_i\bar{C}_i + \bar{A}_i\bar{C}_i \qquad \text{(A.3-6)}$$

This symmetry will be used later in constructing a full parallel adder.

These equations have been implemented using threshold logic techniques in a manner suggested by Kolb.[11] Referring to Table A.3-2 and Eq. A.3-4 it is seen that a carry is generated from the i^{th} position whenever any two or all three of the A_i, B_i, and C_i bits are present. Thus a transistor circuit with a threshold such that the transistor will be driven from a non-conducting to a conducting state whenever two or more inputs are present will produce outgoing carry information. Such a circuit is shown using transistor Q_1 in Fig. A.3.7. Note that if the same voltage levels are used to represent logical 1's and 0's at the input and output, the negation of the carry function is produced.

For generation of the sum bit, a circuit is needed which will go to a conducting state whenever an odd number of inputs is present but will remain non-conducting for an even number of inputs. Such a circuit is shown using transistor Q_2 in Fig. A.3.7. It is achieved using the output of the carry out circuit as an

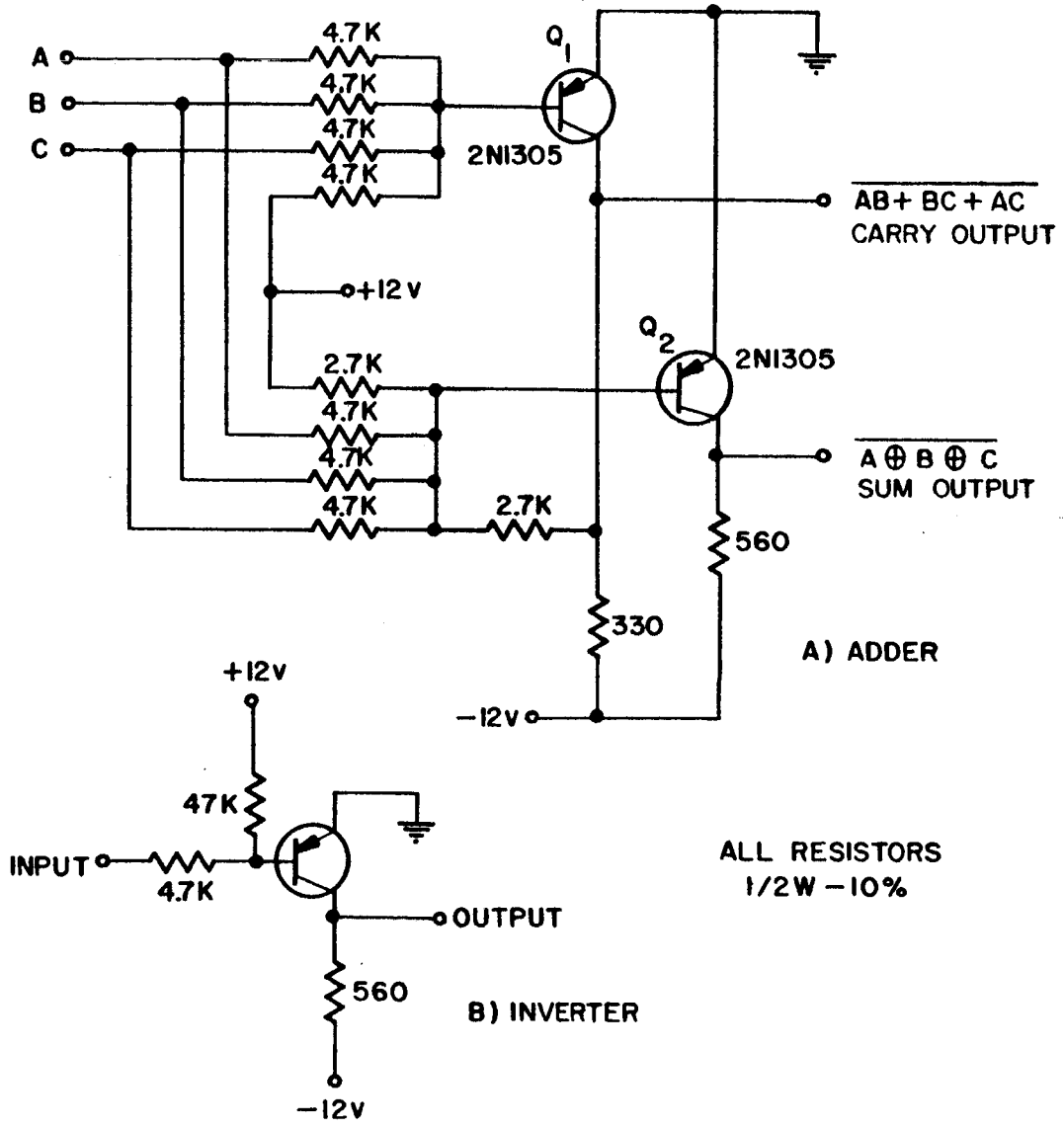


FIGURE A.3.7 – THRESHOLD LOGIC ADDER

additional input. The method of operation is as follows. Transistor Q_2 is non-conducting with no inputs. No carry is being generated. Any one of the inputs A_i, B_i, or C_i can cause it to conduct. If, however, two or more of these inputs are present, a carry is generated. This carry is used to inhibit the transistor from going into the conducting state until all three inputs are present. Again note that if the same voltage levels are used to represent logical 1's and 0's at the input and output, the negation of the sum bit is obtained.

Utilizing the symmetric Eqs. A.3-3 through A.3-6, this basic circuit can be combined to form a full parallel adder as shown in Fig. A.3.8. A minimum carry propagation time is achieved for this logic since the carry only has to pass through one transistor per stage. Alternate sum digits must be inverted but this does not effect the carry propagation times. The inverter circuit used is also shown in Fig. A.3.7.

Since the sum or inverter transistors are not loaded in the threshold circuit, they can be used to drive a voltage ladder type of digital to analog decoder. The decoder used is also shown in Fig. A.3.8.

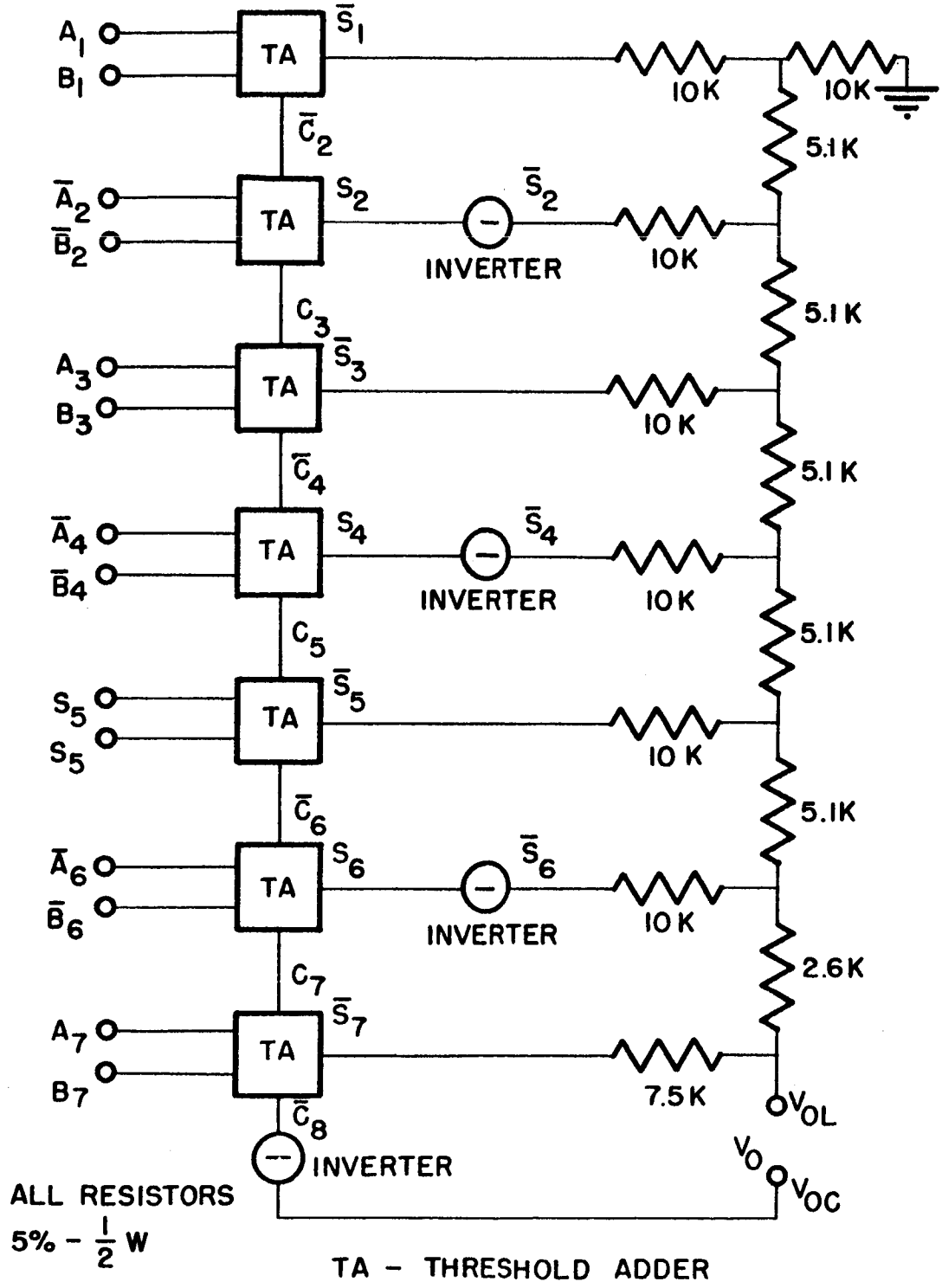


FIGURE A.3.8 — PARALLEL ADDER AND LADDER DECODER

The differential voltage output, V_0, is produced by two component voltages, V_{0L}, the output voltage from the voltage ladder alone, and V_{0C}, the output voltage from the carry.

$$V_0 = V_{0L} - V_{0C}$$

The output of the voltage ladder is

$$V_{0L} = \frac{E_s(p)}{2^n}$$

where E_s is the supply (in this case -12 volts), p is the binary number input to the ladder, and n is the number of stages.[27] The carry output voltage is given by:

$$V_{0C} = E_s C_8$$

where C_8 is the Boolean carry function.

If the minuend is greater than or equal to the subtrahend, $C_8 = 0$ and $V_{0C} = 0$. Then $p = \bar{S}_1\bar{S}_2 \cdots \bar{S}_7 = (2^n - 1 - S)$, and

$$V_{0L} = \frac{E_s}{2^n}(2^n - 1 - S).$$

But,

$$S = A^1 + B = 2^n - (A - B) - 1.$$

So,

$$V_0 = V_{0L} = \frac{E_s}{2^n}(2^n - 1 - 2^n + (A - B) + 1)$$

$$V_0 = \frac{E_s}{2^n}(A - B) \qquad \text{(A.3-7)}$$

Thus the output voltage is proportional to (A - B) for $A \geq B$.

If $A < B$, $C_8 = 1$. Then $p = \bar{S}_1\bar{S}_2 \ldots \bar{S}_7 = 2^n - 1 - (B - A - 1)$.

$$V_0 = V_{0L} - V_{0C} = \left[\frac{E_s}{2^n} 2^n - 1 - (B - A - 1)\right] - E_s$$

$$V_0 = \frac{E_s}{2^n}(A - B) \qquad (A.3\text{-}8)$$

This is the same as Eq. A.3-7, so that a differential voltage output is obtained which is proportional in both magnitude and sign to the difference of the inputs to the comparator. Any drift in the supply affects only the proportionality constant since all stages are driven by one voltage supply.

The differential signal is modulated so that it can be used to drive the 60 cps. ac servomotors of the positioning table. This modulation is accomplished as shown in Fig. A.3.9.

The voltage generated from the C_8 transistor is applied to the center tap of the primary of UTC A19 transformer. The output from the voltage ladder is applied to the wiper of a C.P. Clare HGS - 1004 mercury wetted contact chopper relay. The outputs of the chopper relay are connected to the ends of the transformer primary. Thus the differential voltage is applied alternately across the two halves of the transformer primary, producing a proportional signal in the secondary which has the

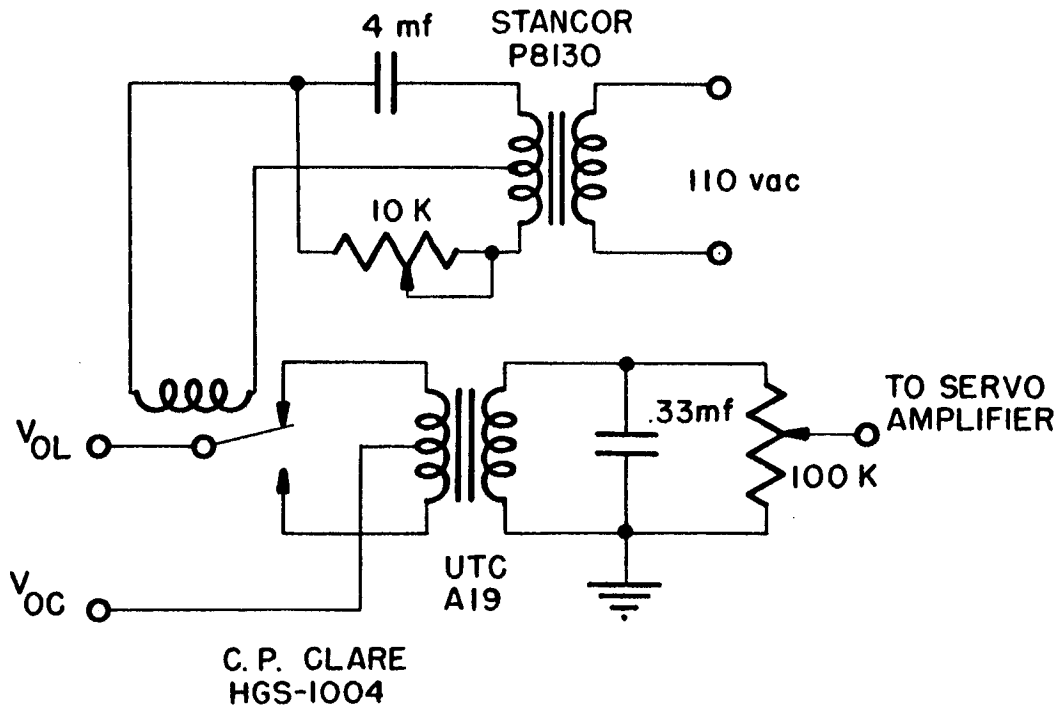


FIGURE A.3.9 — MODULATOR

proper phasing. This signal is filtered before using it to drive the servo amplifier.

The drive to the chopper is provided by a standard filament transformer with some additional phase lead elements which allow proper phasing of the modulated voltage with respect to line voltage.

Servo Amplifiers, Motors, and Gear Trains

The two-phase ac servomotor used was a Diehl Manufacturing Co. No. FPE 49-56-2 15 Watt Low Inertia Servomotor with an ac tachometer and a high impedance control winding. The voltage on the control winding determines the motor speed while its phase determines the direction of motor rotation.

The prime mover amplifier used to supply this motor is shown in Fig. A.3.10. Because of their high impedance, the control windings are supplied directly by a pair of 6L6 tubes in push-pull eliminating the need for an output transformer. The phase of the tachometer feedback can be adjusted by the network shown. The 100K potentiometer controls the feedback gain. The signal from the wiper of this potentiometer is combined in the first stage with the signal from the modulator to produce the difference since the two signals are 180^{o} out of phase. This

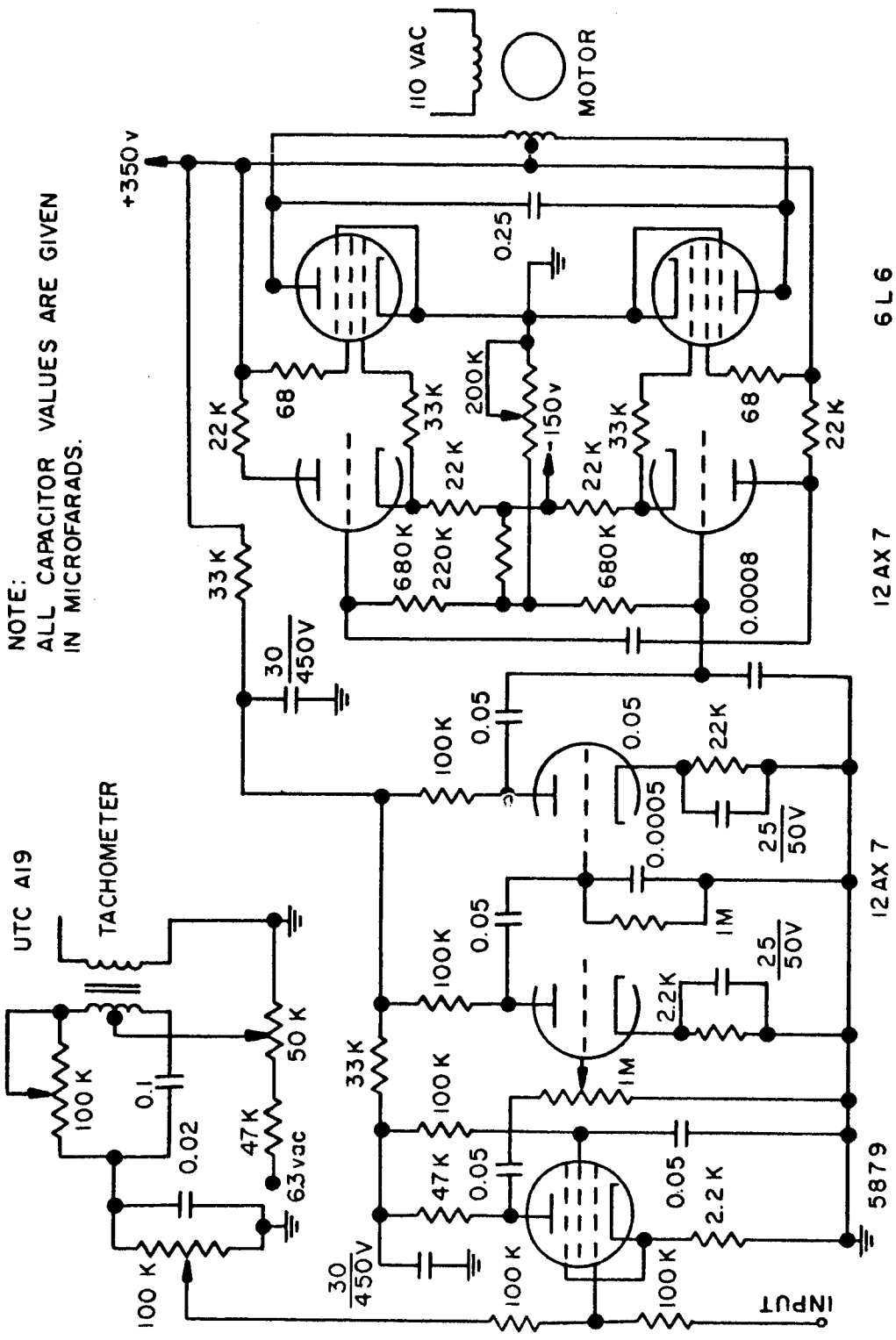


FIGURE A.3.10 SERVO AMPLIFIER

difference is amplified and inverted to drive the output tubes in push-pull operation. A small ac signal is used to bias the center tap of the secondary of the transformer from the tachometer to null out the small zero speed signal.

The two phase ac Diehl servomotor described above was chosen because the prime mover power level is sufficient for the needed positioning. There is no brush friction and the velocity feedback allows the motor time constants to be reduced. The stall torque can be easily handled by the instrument gear trains. Each gear train was breadboarded using three meshes of ratios 5:1, 5:1, and 1.5:1 providing an overall ratio of 37.5:1.

APPENDIX IV

INITIAL CONDITION SAMPLE CALCULATIONS AND CODING TABLES

The initial conditions to the derivative registers at the start of each path segment are calculated using Eqs. 2-28, 2-29 and 2-30. These equations are repeated here for convenience.

$$y'(x_0) = \frac{1}{h}\left(\Delta y_0 - \frac{\Delta^2 y_0}{2} + \frac{\Delta^3 y_0}{3}\right) \qquad (2\text{-}28)$$

$$y''(x_0) = \frac{1}{h^2}\left(\Delta^2 y_0 - \Delta^3 y_0\right) \qquad (2\text{-}29)$$

$$y'''(x_0) = \frac{\Delta^3 y_0}{h^3} \qquad (2\text{-}30)$$

Sample calculations for the initial conditions introduced at point 23 of the closed contour and point 1 of the spiral will now be presented. Also to be presented is the proper tape coding of these conditions.

At point 23 of the closed contour, x is the dependent variable. The values of the differences are $\Delta x_{23} = -1$, $\Delta^2 x_{23} = -1$,

and $\Delta^3 x_{23} = -2$. These are found from Fig. 4.3. Interchange of the roles of x and y in Eqs. 2-28, 2-29 and 2-30 then gives

$$x'(y_{23}) = 2^{-3}(-1 + \frac{1}{2} - \frac{2}{3}) = 2^{-3}(-1\frac{1}{6}),$$

$$x''(y_{23}) = 2^{-6}(-1 + 2) = 2^{-6}, \text{ and}$$

$$x'''(y_{23}) = 2^{-9}(-2)$$

The binary coding of the first derivative entry is then

$$-0.001 \quad 00101010 \quad 10101010$$

which when converted to two's complement form is

$$\underbrace{11.110}\ \underbrace{11010101}\ \underbrace{01010110}$$

To conform with the tape format, this is entered on the three lines of tape used for the first derivative as marked off. The second derivative entry becomes

$$\underbrace{0.000100}\ \underbrace{00000000}\ \times 2^{-2}$$

Recalling that the highest order bit to be entered on the tape is 2^{-3}, this is then entered on the two lines representing the second derivative as marked off. The third derivative entry becomes

$$-.0010000 \times 2^{-5}$$

which when converted to two's complement form is

$$\underbrace{1.1110000}\ \times 2^{-5}$$

This conforms to the tape format and is entered on the third derivative line of the tape. These entries are shown in Block 23 of Fig. 4.5.

At point 1 of the spiral, $\Delta y_0 = -11$, $\Delta^2 y_0 = 2$, and $\Delta^3 y_0 = 0$. These are found from Fig. 4.7. Using the appropriate formulas,

$$y'(x_0) = 2^{-3}(-11 - 1) = 2^{-3}(-12),$$

$$y''(x_0) = 2^{-6}(2) = 2^{-5}, \text{ and}$$

$$y'''(x_0) = 0$$

The binary coding of the entry to the first derivative register is then

$$-1.100\ \ 00000000\ \ 00000000$$

which when converted to two's complement and the proper format gives the entry to the tape lines representing the first derivative as

$$\underbrace{10.100}\ \underbrace{00000000}\ \underbrace{00000000}$$

The coding of the second derivative entry becomes

$$\underbrace{0.001000}\ \underbrace{00000000}\ \times 2^{-2}$$

To conform to the tape format, this is entered as marked off. The coding to the third derivative entry is all 0's. These entries are shown in Block 1 of Fig. 4.8.

The fourth difference corrections to the first and third derivative registers must be coded using the tape format shown in Fig. 4.4. To do this, Tables A.4-1, A.4-2 and A.4-3 are used. The first two tables contain the coding of $-\frac{1}{6h}\ \Delta^4$ (dependent variable) which is the correction term to the first derivative register. The last table contains the tape coding for $\frac{1}{h^3}\Delta^4$ (dependent variable) which is the correction term to the third derivative register. Remember that a straight binary representation is used for positive corrections and a two's complement representation is used for negative corrections.

TABLE A.4-1

CODING TABLE FOR FOURTH DIFFERENCE CORRECTIONS TO THE FIRST DERIVATIVE REGISTER

	Line 1	Line 2	Line 3
$\Delta^4 y$	87654321	87654321	87654321
0	00000000	00000000	00000000
+1	00011111	11010101	01010110
+2	00011111	10101010	10101011
+3	00011111	10000000	00000000
+4	00011111	01010101	01010110
+5	00011111	00101010	10101011
+6	00011111	00000000	00000000
+7	00011110	11010101	01010110
+8	00011110	10101010	10101011
+9	00011110	10000000	00000000
+10	00011110	01010101	01010110
+11	00011110	00101010	10101011
+12	00011110	00000000	00000000
+13	00011101	11010101	01010110
+14	00011101	10101010	10101011
+15	00011101	10000000	00000000

TABLE A. 4-2

CODING TABLE FOR FOURTH DIFFERENCE CORRECTIONS TO THE FIRST DERIVATIVE REGISTER

$\Delta^4 y$	Line 1 87654321	Line 2 87654321	Line 3 87654321
0	00000000	00000000	00000000
-1	00000000	00101010	10101010
-2	00000000	01010101	01010101
-3	00000000	10000000	00000000
-4	00000000	10101010	10101010
-5	00000000	11010101	01010101
-6	00000001	00000000	00000000
-7	00000001	00101010	10101010
-8	00000001	01010101	01010101
-9	00000001	10000000	00000000
-10	00000001	10101010	10101010
-11	00000001	11010101	01010101
-12	00000010	00000000	00000000
-13	00000010	00101010	10101010
-14	00000010	01010101	01010101
-15	00000010	10000000	00000000

TABLE A. 4-3

CODING TABLE FOR FOURTH DIFFERENCE CORRECTIONS TO THE THIRD DERIVATIVE REGISTER

Positive		Negative
8 7 6 5 4 3 2 1	$\Delta^4 y$	8 7 6 5 4 3 2 1
0 0 0 0 0 0 0 0	0	0 0 0 0 0 0 0 0
0 0 0 0 1 0 0 0	1	1 1 1 1 1 0 0 0
0 0 0 1 0 0 0 0	2	1 1 1 1 0 0 0 0
0 0 0 1 1 0 0 0	3	1 1 1 0 1 0 0 0
0 0 1 0 0 0 0 0	4	1 1 1 0 0 0 0 0
0 0 1 0 1 0 0 0	5	1 1 0 1 1 0 0 0
0 0 1 1 0 0 0 0	6	1 1 0 1 0 0 0 0
0 0 1 1 1 0 0 0	7	1 1 0 0 1 0 0 0
0 1 0 0 0 0 0 0	8	1 1 0 0 0 0 0 0
0 1 0 0 1 0 0 0	9	1 0 1 1 1 0 0 0
0 1 0 1 0 0 0 0	10	1 0 1 1 0 0 0 0
0 1 0 1 1 0 0 0	11	1 0 1 0 1 0 0 0
0 1 1 0 0 0 0 0	12	1 0 1 0 0 0 0 0
0 1 1 0 1 0 0 0	13	1 0 0 1 1 0 0 0
0 1 1 1 0 0 0 0	14	1 0 0 1 0 0 0 0
0 1 1 1 1 0 0 0	15	1 0 0 0 1 0 0 0

BIBLIOGRAPHY

1. Brambrut, S. A., "Requirements for a Curvilinear Interpolator Using Incremental Computation", M.S. Thesis, Massachusetts Institute of Technology, 1959.

2. Cowan, R. A., "A Digital Non-linear Function Generator Operating with Absolute Data", Ph.D. Thesis, Case Institute of Technology, 1962.

3. Flores, I., The Logic of Computer Arithmetic. New York: Prentice Hall Inc., 1963, pp. 20-42.

4. Henegar, H. B., "New Continuous Path System Uses DDA Interpolator", Control Engineering, January 1961, pp. 71-76.

5. Hills, F. B., "A Study of Incremental Computation by Difference Equations", M.S. Thesis, Massachusetts Institute of Technology, 1958.

6. Johnson, E. C., "Interpolating Between Programmed Points to Get Smooth Curves", Control Engineering, June 1957, pp. 153-157.

7. Kaiwa, T. and Inaba, S., "The Features of Latest Japanese Numerical Control", Control Engineering, October 1961, pp. 88-91.

8. Kamm, L. J., "Digital Curve Generator", U.S. Patent 2,784,359, March 5, 1957.

9. Kintner, P. M., "A Simple Method of Designing NOR Logic", Control Engineering, February 1963, pp. 77-79.

10. Knight, W. W. and Straub, W. W., "A Machine Tool Director; A Special-Purpose Computer for Use in Numerical Control Systems", Computers in Control, AIEE, September 1961, pp. 31-51.

11. Kolb, E. R., "High Density Optical Information Storage and High Speed Retrieval", Ph.D. Thesis, Case Institute of Technology, 1963, pp. 78-93.

12. Maurer, H. E., "Error Analysis of a Digital Differential Analyzer", E. E. Thesis, Massachusetts Institute of Technology, 1957.

13. Mergler, H. W., Moshos, G. J. and Young, A. E., "Machine Tool Control from a Digital-Analog Computer", IRE PGIE, August 1953, pp. 26-30.

14. Mergler, H. W., "A Digital-Analog Machine Tool Control System", Proceeding of the Western Joint Computer Conference, April 1954, pp. 46-49.

15. Mergler, H. W., "A Numerical Machine Tool Control System Operating from Coded Punched-Paper Tape", Ph. D. Thesis, Case Institute of Technology, 1956.

16. Mergler, H. W., et. al., "Digital Control Systems Engineering", Case Institute of Technology Summer Program Notes, Cleveland, Ohio, 1962.

17. Milne, W. E., Numerical Calculus. Princeton: Princeton University Press, 1949, pp. 1-20.

18. Porter, A. and Stoneman, F., "A New Approach to the Design of Pulse Monitored Servo Systems", Proceedings of IEE, 1950, pp. 597-611.

19. Proceedings of the EIA Symposium on Numerical Control Systems for Machine Tools. New York: The AC Book Company, 1957.

20. Rosenberg, J., "Logical Organization of the Digimatic Computer", Proceedings of the Eastern Joint Computer Conference, December 1957, pp. 25-29.

21. Rosenberg, J., "Automation System", U.S. Patent 2,833,941, May 6, 1958.

22. Rosenberg, J., "Automatic Machine Tool Control", in Leondes, C.T. (ed.), Computer Control Technology. New York: The McGraw-Hill Book Company, 1961, pp. 535-589.

23. Sarachik, P. and Ragazzini, J. R., "A 2-Dimensional Feedback Control System", Transactions of AIEE, Vol. 76, pt. II, 1957, pp. 55-61.

24. Scarborough, J. B., Numerical Mathematical Analysis. Baltimore: The Johns Hopkins Press, 1962, pp. 79-82.

25. Speller, J. and Cooney, J. D., "Interpolating Point-to-Point Inputs for Continuous Machine Control", Control Engineering, June 1957, pp. 114-120.

26. "Stepless Switching Interpolates Punched Card Position Data as Continuous Tool Path Analog", Electrical Manufacturing, October 1957, pp. 116-123.

27. Susskind, A. K. (ed.), Notes on Analog-Digital Conversion Techniques. New York: The Technology Press of Massachusetts Institute of Technology and John Wiley and Sons Inc., 1957, pp. 5.29-5.40.

28. Tysama, S., "Digital Parabolic Interpolator Controls Milling Machine", Control Engineering, December 1961, pp. 82-83.

29. Welch, J. D., "Automatic Feedrate Regulation in Numerically Controlled Contour Milling", M.S. Thesis, Massachusetts Institute of Technology, 1960.

30. Whittaker, E. and Robinson, G., The Calculus of Observations. London: Blackie and Sons, Ltd., 1949, pp. 36-38.